Introduction to Natural Language Processing
HW1: Working with Corpora

This is an easy homework, and it’s a chance to play around a lot and try some stuff in Chaps 1 to 4.

If you don’t already know Python, I suggest you use this week to go through all the sections in the book that are trying to introduce you to Python.
4 Writing Structured Programs

By now you will have a sense of the capabilities of the Python programming language for processing natural language. However, if you're new to Python or to programming, you may still be wrestling with Python and not feel like you are in full control yet. In this chapter we'll address the following questions:

1. How can you write well-structured, readable programs that you and others will be able to re-use easily?
2. How do the fundamental building blocks work, such as loops, functions and assignment?
3. What are some of the pitfalls with Python programming and how can you avoid them?

Along the way, you will consolidate your knowledge of fundamental programming constructs, learn more about using features of the Python language in a natural and concise way, and learn some useful techniques in visualizing natural language data. As before, this chapter contains many examples and exercises (and as before, some exercises introduce new material). Readers new to programming should work through them carefully and consult other introductions to programming if necessary; experienced programmers can quickly skim this chapter.

In the other chapters of this book, we have organized the programming concepts as dictated by the needs of NLP. Here we revert to a more conventional approach where the material is more closely tied to the structure of the programming language. There's not room for a complete presentation of the language, so we'll just focus on the language constructs and idioms that are most important for NLP.

4.1 Back to the Basics

Assignment

Assignment would seem to be the most elementary programming concept, not deserving a separate discussion. However, there are some surprising subtleties here. Consider the following code fragment:

```python
>>> foo = 'Monty'
>>> bar = foo
>>> foo = 'Python'
>>> bar
'Monty'
```
- NLTK comes with lots of corpora
- Corpora may have structure & annotations
from nltk.book import *

Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.

>>> text1
Moby Dick by Herman Melville 1851

>>> text1.concordance("monstrous")
Displaying 11 of 11 matches:
ong the former , one was of a most monstrous size . ... This came towards us ,
ON OF THE PSALMS . " Touching that monstrous bulk of the whale or ork we have r
ll over with a heathenish array of monstrous clubs and spears . Some were thick

d as you gazed , and wondered what monstrous cannibal and savage could ever hav
that has survived the flood ; most monstrous and most mountainous ! That Himmel
they might scout at Moby Dick as a monstrous fable , or still worse and more de
Get some data, count some stuff

- Use the `nltk.FreqDist` class

```python
>>> from nltk.corpus import gutenberg
>>> words_per_sentence = [gutenberg.words(fileid) for fileid in gutenberg.fileids()]

>>> words = [word.lower() for sublist in words_per_sentence for word in sublist]

>>> fdist = nltk.FreqDist(words)
>>> fdist.N()
2621613
>>> fdist.B()
42339
>>> fdist.hapaxes()
['logically', 'sluing', 'compactly', 'purtenance', 'unmodernized', 'trivially',
'dialect', 'gangrene', 'reunited', 'discontinuous', 'vivifies', 'reprinted', 'cl
arions', 'unblamed', 'spectrally', 'aesthetic', 'apportioned', 'rebuker', 'clapp
ings', 'shimeon', 'throttling', 'harum', 'beulah', 'gloat', 'humbleness', 'rhoda

>>> fdist.most_common(10)
[(',', 186091), ('the', 133583), ('and', 95442), ('.', 73746), ('of', 71267), ('to', 48057), (':', 47406), ('a', 33960), ('in', 33580), ('i', 30265)]
A Crow was sitting on a branch of a tree with a piece of cheese in her beak when a Fox observed her and set his wits to work to discover some way of getting the cheese.

"Coming and standing under the tree he looked up and said, "What a noble bird I see above me!", "Her beauty is without equal, the hue of her plumage exquisite.", "If only her voice is as sweet as her looks are fair, she ought without doubt to be Queen of the Birds."", "The Crow was hugely flattered by this, and just to show the Fox that she could sing she gave a loud caw.", "Down came the cheese, of course, and the Fox, snatching it up, said, "You have a voice, madam, I see: what you want is wits.""
Working With Your Own Data

- Reading from a file

```python
>>> print (fables_text[::170])
A Crow was sitting on a branch of a tree with a piece of cheese in her beak when a Fox observed her and set his wits to work to discover some way of getting the cheese.
```
Sentence Detection

- **Sentence:**
  - Something ending with a .. ?, ! (and sometime also :)  
  - “You reminded me,” she remarked, “of your mother.”
    - Nested sentences
    - Note the .”

- Before tokenizing the text into words, we need to segment it into sentences.
Sentence Detection

- NLTK uses the *Punkt* sentence segmenter
- Most ambiguities are from abbreviations
- Uses 3 simple rules to identify them
  - look for very tight collocation consisting of a truncated word and a final period
  - abbreviations are usually short
  - abbreviations can contain internal periods
Tokenization

- Divide text into units called tokens (words, numbers, punctuations)

- What is a word?
  - Graphical word (a token)
    - Dependent on surface properties of text
    - Defined by the company it keeps
  - Underlying word (stem, root...)
    - Dependent on some form of morphological analysis

- Practical definition:
  - an indivisible (!) sequence of characters
  - carries elementary meaning
  - is reusable in different contexts
Tokenization

- Whitespace does not always indicate a word break
- Punctuation
  - “You reminded me,” she remarked
  - O’Hara vs. John’s
- Compound words
  - San Francisco
  - The New York-New Heaven railroad
  - Wake up, work out
    - I couldn’t work the answer out
- Contractions
  - can’t, won’t, etc.
- Merged words
  - Wanna, shoulda, etc.
Today was a very eventful work day. Today was the start of the G20 summit. It happens every year and it is where 20 of the leaders of the world come together to talk about how to run their governments effectively and what not. Since there are so many leaders coming together their are going to be a lot of people who have different views on how to run the government they follow so they protest. There was a protest that happened along the street where I work and at first it looked peaceful until a bunch of people started rebelling and creating a riot. Police cars were burned and things were thrown at cops. Police were in full riot gear to alleviate the violence. As things got worse tear gas and bean bag bullets were fired at the rioters while they smash windows of stores. And this all happened right in front of my store which was kind of scary but it was kind of interesting since I've never seen a riot before.
Tokenization

- Simplest tokenization by whitespace

```python
>>> for sent in sents:
    print sent.split()
...
['Saturday', 'June', '26', '2010', 'eventful', 'it', 'is', 'Today', 'was', 'a', 'very', 'eventful', 'work', 'day.]
['Today', 'was', 'the', 'start', 'of', 'the', 'G20', 'summit.]
['It', 'happens', 'every', 'year', 'and', 'it', 'is', 'where', '20', 'of', 'the', 'leaders', 'of', 'the', 'world', 'come', 'together', 'to', 'talk', 'about', 'how', 'to', 'run', 'their', 'governments', 'effectively', 'and', 'what', 'not.]
['Since', 'there', 'are', 'so', 'many', 'leaders', 'coming', 'together', 'their', 'are', 'going', 'to', 'be', 'a', 'lot', 'of', 'people', 'who', 'have', 'different', 'views', 'on', 'how', 'to', 'run', 'the', 'government', 'they', 'follow', 'so', 'they', 'protest.]
['There', 'was', 'a', 'protest', 'that', 'happened', 'along', 'the', 'street', 'where', 'I', 'work', 'and', 'at', 'first', 'it', 'looked', 'peaceful', 'until', 'a', 'bunch', 'of', 'people', 'started', 'rebellings', 'and', 'creating', 'a', 'riot.]
['Police', 'cars', 'were', 'burned', 'and', 'things', 'were', 'thrown', 'at', 'cops.]
['Police', 'were', 'in', 'full', 'riot', 'gear', 'to', 'alleviate', 'the', 'violence.]
['As', 'the', 'rioters', 'while', 'they', 'smash', 'windows', 'of', 'stores.]
['And', 'this', 'all', 'happened', 'right', 'in', 'front', 'of', 'my', 'store', 'which', 'was', 'kind', 'of', 'scary', 'but', 'it', 'was', 'kind', 'of', 'interesting', 'since', 'I\'ve', 'never', 'seen', 'a', 'riot', 'before.]
['Since', 'it', 'all', 'happened', 'in', 'front', 'of', 'close', 'to', 'my', 'store', 'my', 'coworkers', 'and', 'I', 'were', 'stuck', 'in', 'the', 'store', 'until', 'it', 'was', 'safe', 'to', 'come', 'out', 'so', 'we', 'saw', 'everything', 'happen', 'from', 'start', 'to', 'finish.]
['Sucks', 'that', 'everything', 'they', 'were', 'protesting', 'is', 'now', 'a', 'lost', 'cause', 'since', 'everyone', 'will', 'just', 'remember', 'the', 'riot', 'and', 'violence', 'they\'re created.]
['However', 'I\'ve', 'bet', 'the', 'police', 'wanted', 'this', 'to', 'happen', 'just', 'to', 'justify', 'that', 'the', 'money', 'they\'re spent', 'for', 'the', 'extra', 'security', 'was', 'well', 'worth', 'it.]
['Posted', 'by']
```
Some Problems

- What if we tried to count the word frequencies?

<table>
<thead>
<tr>
<th>Word</th>
<th>Frequency</th>
<th>Word</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>26,</td>
<td>1</td>
<td>Sucks</td>
<td>1</td>
</tr>
<tr>
<td>And</td>
<td>1</td>
<td>There</td>
<td>1</td>
</tr>
<tr>
<td>As</td>
<td>1</td>
<td>about</td>
<td>1</td>
</tr>
<tr>
<td>G20</td>
<td>1</td>
<td>alleviate</td>
<td>1</td>
</tr>
<tr>
<td>However,</td>
<td>1</td>
<td>along</td>
<td>1</td>
</tr>
<tr>
<td>I've</td>
<td>1</td>
<td>bag</td>
<td>1</td>
</tr>
<tr>
<td>It</td>
<td>1</td>
<td>be</td>
<td>1</td>
</tr>
<tr>
<td>June</td>
<td>1</td>
<td>bean</td>
<td>1</td>
</tr>
<tr>
<td>Posted</td>
<td>1</td>
<td>before.</td>
<td>1</td>
</tr>
<tr>
<td>Saturday,</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tokenization

- Other (better) approaches
  - Define a set of rules to split punctuation
  - Use machine learning to identify word boundaries
  - Look for reoccurring patterns in large corpora

- Default tokenizer in NLTK uses option 3 (*Punkt*)
  - Uses distributional definition of words to identify boundaries
  - Similar to the sentence delimiting module
Conditional Counts

- We can also get some more interesting information by using a `ConditionalFreqDist`
- How often have I seen word$_2$ given that word$_1$ immediately preceded it?
- *fox* is seen exactly twice after having seen *the*
Ngrams & Counting Other Things

- Can basically count anything with FreqDist

- N-grams are sequences of $n$ consecutive words e.g., "more is said than done"
  - Unigrams: "more", "is", "said", "than", "done"
  - Bigrams: "more is", "is said", "said than", "than done"
  - Trigrams: "more is said", "is said than", "said than done"
  - ...

- Used a lot in NLP applications
  - Language models (next week)
  - Collocation (next)
  - Language Identification
  - ASR
  - Machine Translation
N-grams & POS tagging NLTK

- Built in method nltk.bigrams
- For larger contexts use nltk.util.ngrams(words, n)
- From previous example we could count the bigrams using:
  
  bgdist = nltk.FreqDist(nltk.bigrams(words))
Collocations
Collocations: Definition

“Collocations ... are statements of the habitual or customary places of [a] word.” (Firth 1957)

Usually, we specify an **n-gram** window within which to analyse collocations:
- bigram: credit card, credit crunch
- trigram: credit card fraud, credit card expiry
  - ...

The idea is to look at co-occurrence of words within a specific n-gram window

**Characteristics/Expectations:**
- regular/frequently attested
- occur within a narrow window (span of few words)
- not fully compositional
- non-substitutable
- subject to category restrictions
A Motivating Example

Consider phrases such as:
- strong tea ? powerful tea
- strong support ? powerful support
- powerful drug ? strong drug
- powerful computers ? strong computers

Traditional semantic theories have difficulty accounting for these patterns.
- *strong* and *powerful* seem near-synonyms
- do we claim they have different senses?
- what is the crucial difference?
- why do *big* and *large* have different frequencies depending on the noun they modify?
Regularity / Frequency

- \( f(\text{strong tea}) > f(\text{powerful tea}) \)

- \( f(\text{credit card}) > f(\text{credit bankruptcy}) \)

- \( f(\text{white wine}) > f(\text{yellow wine}) \)
  - (even though white wine is actually yellowish)
Narrow Window (Textual Proximity)

- Usually collocates of a word occur close to that word.
  - may still occur across a span

- Examples:
  - bigram: *white wine*, *powerful tea*
  - >bigram: *knock on the door*; *knock on X’s door*

- Can count n-grams with intervening words:
  - *federal (.* ) subsidy*
  - matches: *federal subsidy*, *federal farm subsidy*, *federal manufacturing subsidy*...
Non-Compositionality

- *white wine*
  - not really “white”, meaning not fully predictable from component words + syntax

- Similarly:
  - *heavy rain* (not *big rain* or *powerful rain*)
  - *regression coefficient*
  - *good practice guidelines*

- Extreme cases:
  - idioms such as *kick the bucket*
  - meaning is completely frozen
Non-Substitutability

- If a phrase is a collocation, we can’t substitute a word in the phrase for a near-synonym, and still have the same overall meaning.

- E.g.:
  - white wine vs. yellow wine
  - powerful tea vs. strong tea
  - Big rain vs. heavy rain
Category Restrictions

- Frequency alone doesn’t indicate collocational strength:
  - *by the* is a very frequent phrase in English
  - not a collocation
  - Also see [http://en.wikipedia.org/wiki/Collocation](http://en.wikipedia.org/wiki/Collocation)

- Collocations tend to be formed from content words:
  - A+N: *powerful tea*
  - N+N: *regression coefficient, mass demonstration*
  - N+PREP+N: *degrees of freedom*
Importance of Collocations

1. Several applications need to “know” about collocations:
   - **terminology extraction**: technical or domain-specific phrases crop up frequently in text (oil prices)
   - **document classification**: specialist phrases are good indicators of the topic of a text
   - **named entity recognition**: names such as New York tend to occur together frequently; phrases like new toy don’t
Collocations in NLTK

- Use the nltk.Text module

```python
from nltk.corpus import gutenberg
words_per_sentence = [gutenberg.words(fileid) for fileid in gutenberg.fileids()]
words = [word.lower() for sublist in words_per_sentence for word in sublist]
gutext = nltk.Text(words)
gutext.collocations()
```

```python
>>> gutext.collocations()
thou shalt; said unto; thou hast; thus saith; thou art; captain wentworth; lord god; frank churchill; unto thee; every one; sperm whale; burnt offering; jesus christ; lady russell; colonel brandon; say unto; miss woodhouse; father brown; spake unto; buster bear
```
The Empiricist's View of Meaning

- Firth's view (1957):
  - “You shall know a word by the company it keeps”

- This is a contextual view of meaning, akin to that espoused by Wittgenstein (1953).
  - i.e., meaning is how it's used

- In the Firthian tradition, attention is paid to patterns that crop up with regularity in language.

- Statistical work on collocations tends to follow this tradition.
Another View of Lexical Categories

- Word categories can be defined by the context in which they appear.
- For a word $w$ find all the contexts $w_1w_2$ in which $w$ appears.
- Find all words $w'$ that share many frequent contexts.
Finding Similar Words in NLTK

- Use the `nltk.Text` class
- Use the similar function.

```python
>>> gutext
<Text: emma by jane austen 1816>
>>> gutext.similar('woman')
man people time day thing men one king place lord house night land word other earth way lady world gentleman
```
Stemming
Morphology

- Words can have compositional meaning from their parts
- Morphology is the study of the internal structure of words, of the way words are built up from smaller meaning units.

Morpheme:
- The smallest meaningful unit in the grammar of a language.

Two classes of morphemes
- Stems: “main” morpheme of the word, supplying the main meaning (i.e. establish in the example below)
- Affixes: add additional meaning
  - Prefixes: Antidisestablishmentarianism
  - Suffixes: Antidisestablishmentarianism
  - Infixes: hingi (borrow) – humingi (borrower) in Tagalog
  - Circumfixes: sagen (say) – gesagt (said) in German
Stemming

• The removal of the inflectional ending from words (strip off any affixes)
  • Laughing, laugh, laughs, laughed $\rightarrow$ laugh

• Problems
  • Can conflate semantically different words
    • Gallery and gall may both be stemmed to gall

• A further step is to make sure that the resulting form is a known word in a dictionary, a task known as lemmatization.
NLTK Stemmers

- nltk.wordnet.morphy
- A slightly more sophisticated approach
- Use an understanding of inflectional morphology
  - Use an Exception List for irregulars
  - Handle collocations in a special way
- Do the transformation, compare the result to the WordNet dictionary
- If the transformation produces a real word, then keep it, else use the original word.
- For more details, see
  - http://wordnet.princeton.edu/man/morphy.7WN.html
Is Stemming Useful

- For information retrieval, some improvement for smaller documents
  - Helps a lot for some queries, hurts a lot in other cases

- Mixed results for language modeling

- Problems
  - Word sense disambiguation on query terms: *business* may be stemmed to *busy, saw* (the tool) *to see*
  - A truncated stem can be unintelligible to users

- **However**, finding the root word (lemma) may be necessary to use lexical resources
Text Normalization

- Stemming

- Convert to lower case

- Identifying *non-standard words* including numbers, abbreviations, and dates, and mapping any such tokens to a special vocabulary.
  - For example, every decimal number could be mapped to a single token 0.0, and every acronym could be mapped to AAA. This keeps the vocabulary small and improves the accuracy of many tasks.

- Lemmatization / Stemming
  - Make sure that the resulting form is a known word in a dictionary
Moving Beyond Words: Part-of-Speech
Part-of-Speech

- Sometimes lexical items are too specific
  - Don't generalize well
  - Sparse data problems
- Grouping words into a small set of word classes or lexical categories can be useful for many applications
- Part-of-speech is a common method of categorizing words
  - Grounded in linguistic theory
Useful Applications

- Word sense disambiguation
  - The plane **banked** right.
  - We put our money in the **bank**.
- Categorizing genre's of text
  - Fiction vs. Scientific writing
- Identifying writer styles or personality
- Used in downstream processes in the NLP pipeline
  - E.g., syntactic parsing
- Lot's of other applications
```python
>>> import nltk
>>> 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')]
```

```python
>>> nltk.help.upenn_tagset('RB')
RB:  adverb
    occasionally unabatingly maddeningly adventurously professedly
    stirringly prominently technologically magisterially predominately
    swiftly fiscally pitilessly ...
```
Part-Of-Speech (POS)

- Hard to define exactly and no absolute agreement on definition, but it is broadly...

- "a linguistic category of words (or more precisely lexical items), which is generally defined by the syntactic or morphological behaviour of the lexical item in question" – Wikipedia

- 8 traditional POS tags in English
  - Noun, pronoun, adjective, verb, adverb, preposition, conjunction, interjection

- Can be further subcategorized by function
Part-Of-Speech (POS)

- 8 traditional POS tags in English
  - Noun, pronoun, adjective, verb, adverb, preposition, conjunction, interjection

- I talk.
- I am talking.
- I talk slowly.
- I talk to the cats.
- I talk to the cats in the garden.
- Oh! I love talking gleefully to the cats and the birds in the garden.
Tagsets

- The specific set of POS tags used for an application is called the tagset.

- Most popular is the Penn Treebank Tagset:
  - 36 tags
  - [Link to Penn Treebank Tagset](https://www.comp.leeds.ac.uk/ccalas/tagsets/upenn.html)
  - Be wary of funny names for POS:
    - *IN* is a preposition, *JJ* is an adjective, *NN* is a noun (kind of makes sense), *NNS* is a plural noun, etc.

- In contrast the Brown Tagset has 85 POS categories.
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>CC</td>
<td>Coordinating conjunction</td>
</tr>
<tr>
<td>2.</td>
<td>CD</td>
<td>Cardinal number</td>
</tr>
<tr>
<td>3.</td>
<td>DT</td>
<td>Determiner</td>
</tr>
<tr>
<td>4.</td>
<td>EX</td>
<td>Existential <em>there</em></td>
</tr>
<tr>
<td>5.</td>
<td>FW</td>
<td>Foreign word</td>
</tr>
<tr>
<td>6.</td>
<td>IN</td>
<td>Preposition/subordinating conjunction</td>
</tr>
<tr>
<td>7.</td>
<td>JJ</td>
<td>Adjective</td>
</tr>
<tr>
<td>8.</td>
<td>JJR</td>
<td>Adjective, comparative</td>
</tr>
<tr>
<td>9.</td>
<td>JJ$S$</td>
<td>Adjective, superlative</td>
</tr>
<tr>
<td>10.</td>
<td>LS</td>
<td>List item marker</td>
</tr>
<tr>
<td>11.</td>
<td>MD</td>
<td>Modal</td>
</tr>
<tr>
<td>12.</td>
<td>NN</td>
<td>Noun, singular or mass</td>
</tr>
<tr>
<td>13.</td>
<td>NNS</td>
<td>Noun, plural</td>
</tr>
<tr>
<td>14.</td>
<td>NNP</td>
<td>Proper noun, singular</td>
</tr>
<tr>
<td>15.</td>
<td>NNPS</td>
<td>Proper noun, plural</td>
</tr>
<tr>
<td>16.</td>
<td>PDT</td>
<td>Predeterminer</td>
</tr>
<tr>
<td>17.</td>
<td>POS</td>
<td>Possessive ending</td>
</tr>
<tr>
<td>18.</td>
<td>PRP</td>
<td>Personal pronoun</td>
</tr>
<tr>
<td>19.</td>
<td>PP$</td>
<td>$</td>
</tr>
<tr>
<td>20.</td>
<td>RB</td>
<td>Adverb</td>
</tr>
<tr>
<td>21.</td>
<td>RBR</td>
<td>Adverb, comparative</td>
</tr>
<tr>
<td>22.</td>
<td>RBS</td>
<td>Adverb, superlative</td>
</tr>
<tr>
<td>23.</td>
<td>RP</td>
<td>Particle</td>
</tr>
<tr>
<td>24.</td>
<td>SYM</td>
<td>Symbol (mathematical or scientific)</td>
</tr>
<tr>
<td>25.</td>
<td>TO</td>
<td>to</td>
</tr>
<tr>
<td>26.</td>
<td>UH</td>
<td>Interjection</td>
</tr>
<tr>
<td>27.</td>
<td>VB</td>
<td>Verb, base form</td>
</tr>
<tr>
<td>28.</td>
<td>VBD</td>
<td>Verb, past tense</td>
</tr>
<tr>
<td>29.</td>
<td>VBG</td>
<td>Verb, gerund/present participle</td>
</tr>
<tr>
<td>30.</td>
<td>VBN</td>
<td>Verb, past participle</td>
</tr>
<tr>
<td>31.</td>
<td>VBP</td>
<td>Verb, non-3rd ps. sing. present</td>
</tr>
<tr>
<td>32.</td>
<td>VBZ</td>
<td>Verb, 3rd ps. sing. present</td>
</tr>
<tr>
<td>33.</td>
<td>WDT</td>
<td><em>wh</em>-determiner</td>
</tr>
<tr>
<td>34.</td>
<td>WP</td>
<td><em>wh</em>-pronoun</td>
</tr>
<tr>
<td>35.</td>
<td>WP$</td>
<td>$</td>
</tr>
<tr>
<td>36.</td>
<td>WRB</td>
<td><em>wh</em>-adverb</td>
</tr>
<tr>
<td>37.</td>
<td>#</td>
<td>Pound sign</td>
</tr>
<tr>
<td>38.</td>
<td>$</td>
<td>Dollar sign</td>
</tr>
<tr>
<td>39.</td>
<td>.</td>
<td>Sentence-final punctuation</td>
</tr>
<tr>
<td>40.</td>
<td>,</td>
<td>Comma</td>
</tr>
<tr>
<td>41.</td>
<td>:</td>
<td>Colon, semi-colon</td>
</tr>
<tr>
<td>42.</td>
<td>(</td>
<td>Left bracket character</td>
</tr>
<tr>
<td>43.</td>
<td>)</td>
<td>Right bracket character</td>
</tr>
<tr>
<td>44.</td>
<td>”</td>
<td>Straight double quote</td>
</tr>
<tr>
<td>45.</td>
<td>‘</td>
<td>Left open single quote</td>
</tr>
<tr>
<td>46.</td>
<td>”</td>
<td>Left open double quote</td>
</tr>
<tr>
<td>47.</td>
<td>’</td>
<td>Right close single quote</td>
</tr>
<tr>
<td>48.</td>
<td>”</td>
<td>Right close double quote</td>
</tr>
</tbody>
</table>
Part-of-Speech Tagging

- Assign each word in continuous text a tag indicating its part of speech.
  - Essentially a classification problem (later in the course)

- Current state of the art:
  - taggers typically have 96-97% accuracy
  - figure evaluated on a per-word basis
  - in a corpus with sentences of average length 20 words, 96% accuracy can mean one tagging error per sentence
Sources of difficulty

- Mostly due to ambiguity when words have more than one possible tag.
  - need context to make a good guess about POS
  - context alone won’t suffice

- A simple approach which assigns only the most common tag to each word performs with 90% accuracy!
Some Features for Automatic Tagging

- **Syntagmatic information**: the tags of other words in the context of $w$
  - Not sufficient on its own.

- **Lexical information** (“dictionary”): most common tag(s) for a given word *(a bit later today)*
  - e.g. in English, many nouns can be used as verbs *(flour the pan, wax the car...)*
  - however, their most likely tag remains NN
  - distribution of a word’s usages across different POSs is uneven: usually, one highly likely, other much less

- **Other lexical features**
  - Is it uppercase?
  - Prefix, suffix?
Part-of-Speech Tagging

- Assign each word in continuous text a tag indicating its part of speech.
  - Essentially a classification problem (later in the course)

- Current state of the art (varies by corpus):
  - taggers typically have 96-97% accuracy
  - figure evaluated on a per-word basis
  - in a corpus with sentences of average length 20 words, 96% accuracy can mean one tagging error per sentence
Examples of USING bigrams

```python
words = [nltk.word_tokenize(sentence) for sentence in sentences]
flat_words = [word.lower() for sentence in words for word in sentence]

>>> bigrams = nltk.bigrams(flat_words)
>>> bigrams
<generator object bigrams at 0x102c9f288>
>>> bgdist.most_common(50)
[('(', ',', 'and'), 2), ('crow', 'was', 2), ('the', 'fox', 2), ('.', '````', 2), ('the', 'cheese', 2), ('said', ',', 2), ('i', 'see', 2), ('see', '``'), ('just', 'to', 1), ('and', 'just', 1), ('i', 'i', 1), ('her', '1'), ('a', 'ble', 1), ('of', 1), ('as', 'sweet', 1), ('hugely', 'flattered', 1), ('voice', 1), ('fair', '1'), ('is', 'wits', 1), ('what', 'a', 1), ('of', 'the', 1), ('her', 'beak', 1), ('and', 'set', 1), ('the', 'hue', 1), ('work', 'to', 1), ('as', 'her', 1), ('without', 'doubt', 1), ('piece', 'of', 1), ('up', '1'), ('sitting', 1), ('down', 'came', 1), ('of', 'her', 1), ('when', 'a', 1), ('over', 'some', 1), ('looked', 'up', 1), ('``````', 'you', 1), ('a', 'piece', 1), ('ve', 'me', 1), ('me', '!', 1), ('wits', '...', 1), ('by', 'this', 1), ('snatching', 'it', 1), ('could', 'sing', 1), ('sing', 'she', 1), ('caw', '1'), ('voice', '1), ('and', 'standing', 1), ('cheese', '1'), ('some', 'way', 1)]
```
Example of USING pos_tag

>>> nltk.pos_tag(flat_words)
[('a', 'DT'), ('crow', 'NN'), ('was', 'VBD'), ('sitting', 'VBG'), ('on', 'IN'), ('a', 'DT'), ('branch', 'NN'), ('of', 'IN'), ('a', 'DT'), ('tree', 'NN'), ('with', 'IN'), ('a', 'DT'), ('piece', 'NN'), ('of', 'IN'), ('cheese', 'JJ'), ('in', 'IN'), ('her', 'PRP$'), ('beak', 'NN'), ('when', 'WRB'), ('a', 'DT'), ('fox', 'NN'), ('observed', 'VBN'), ('her', 'PRP$'), ('and', 'CC'), ('set', 'NN'), ('his', 'PRP$'), ('wits', 'NNS'), ('to', 'TO'), ('work', 'VB'), ('to', 'TO'), ('discovering', 'VB'), ('some', 'DT'), ('way', 'NN'), ('of', 'IN'), ('getting', 'VBG'), ('the', 'DT'), ('cheese', 'JJ'), ('off', 'IN'), ('standing', 'NN'), ('under', 'IN'), ('the', 'DT'), ('tree', 'NN'), ('he', 'PRP'), ('looked', 'VBD'), ('up', 'RP'), ('and', 'CC'), ('said', 'VBD'), ('what', 'WP'), ('a', 'DT'), ('noble', 'JJ'), ('bird', 'NN'), ('i', 'PRP'), ('see', 'VBP'), ('above', 'IN'), ('me', 'PRP'), ('!', ''''), ('her', 'PRP$'), ('beauty', 'NN'), ('is', 'VBZ'), ('without', 'IN'), ('equal', 'JJ'), ('the', 'DT'), ('hue', 'NN'), ('of', 'IN'), ('her', 'PRP$'), ('plumage', 'NN'), ('exquisite', 'NN'), ('if', 'IN'), ('only', 'RB'), ('her', 'PRP$'), ('voice', 'NN'), ('is', 'VBZ'), ('as', 'RB'), ('sweet', 'J'), ('as', 'IN'), ('noble', 'JJ'), ('are', 'VBP'), ('fair', 'JJ'), ('she', 'PRP'), ('ought', 'MD'), ('without', 'VB'), ('doubt', 'NN'), ('to', 'TO'), ('queen', 'VBN'), ('of', 'IN'), ('the', 'DT'), ('birds', 'NN'), ('only', 'RB'), ('the', 'DT'), ('crow', 'NN'), ('was', 'VBD'), ('hugely', 'RB'), ('flatter ed', 'VBN'), ('by', 'IN'), ('this', 'DT'), ('and', 'CC'), ('just', 'RB'), ('to', 'TO'), ('show', 'VB'), ('the', 'DT'), ('fox', 'NN'), ('that', 'IN'), ('she', 'PRP'), ('could', 'MD'), ('sing', 'VB'), ('gave', 'VBD'), ('a', 'DT'), ('loud', 'NN'), ('caw', 'NN'), ('down', 'IN'), ('came', 'VBD'), ('the', 'DT'), ('cheese', 'NN'), ('of', 'IN'), ('course', 'NN'), ('and', 'CC'), ('the', 'DT'), ('fox', 'NN'), ('snatching', 'VBG'), ('it', 'PRP'), ('up', 'RP'), ('said', 'VBD'), ('you', 'PRP'), ('have', 'VBP'), ('a', 'DT'), ('voice', 'NN'), ('madam', 'NN'), ('i', 'PRP'), ('see', 'VBP'), ('what', 'WP'), ('you', 'PRP'), ('want', 'VBP'), ('is', 'VBZ'), ('wits', 'NNS')]
Have covered HW 1.

Homework 2 is going to have a lot more to do in it.

Pay attention to the READING, read before class
Lexicons & Lexical Semantics
Chap 2 NLPP
Sec. 2.4, 2.5, 2.6
Lexical Resources

- A lexicon, or lexical resource, is a collection of words and/or phrases along with associated information such as part of speech and sense definitions.

- A vocabulary (list of words in a text) is the simplest lexical resource.

- Lexical entry
  - A **lexical entry** typically consists of a **headword** (also known as a **lemma**) along with additional information such as the part of speech and the sense definition.
Different types of Lexical Dictionaries

- LIWC: Linguistic Inquiry and Word Count: categorizes words into a hierarchical set of lexical categories (for sentiment classification as well as others)

- Sentiment Lexicons: Classify words into positive and negative polarity, either binary or scalar

- Wordnet:
  - classifies words hierarchically according to an ontology of things in the world, e.g. cup ISA container
  - Tells you the different senses of a word and groups words with their synonyms “service”

- Verbnet:
  - groups verbs by their meaning into an ontology
  - Tells you how verbs ‘subcategorize’ for their arguments
Wordnet and Verbnet in NLTK

Has a great API that let’s you do lots of stuff
Incredibly useful.
<table>
<thead>
<tr>
<th>Directory</th>
<th>Date</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>treebank</td>
<td>Apr 8, 2014 9:30 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>treebank.zip</td>
<td>Apr 8, 2014 9:30 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>udhr</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>udhr.zip</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>udhr2</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>udhr2.zip</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>unicode_samples</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>unicode_samples.zip</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>verbnet</td>
<td>Apr 7, 2014 2:02 PM</td>
<td>Folder</td>
</tr>
<tr>
<td>verbnet.zip</td>
<td>Apr 7, 2014 2:02 PM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>webtext</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>webtext.zip</td>
<td>Apr 8, 2014 9:31 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>wordnet</td>
<td>Apr 7, 2014 2:02 PM</td>
<td>Folder</td>
</tr>
<tr>
<td>wordnet_ic</td>
<td>Apr 8, 2014 9:33 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>wordnet_ic.zip</td>
<td>Apr 8, 2014 9:32 AM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>wordnet.zip</td>
<td>Apr 7, 2014 2:02 PM</td>
<td>ZIP archive</td>
</tr>
<tr>
<td>words</td>
<td>Apr 8, 2014 9:33 AM</td>
<td>Folder</td>
</tr>
<tr>
<td>words.zip</td>
<td>Apr 8, 2014 9:33 AM</td>
<td>ZIP archive</td>
</tr>
</tbody>
</table>
WordNet

- **WordNet** is a large lexicon of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept*.

- Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser.

- NLTK includes the English WordNet, with 155,287 words and 117,659 synonym sets.

**Senses and Synonyms**

- Consider the 2 sentences:
  - *Benz is credited with the invention of the motorcar*
  - *Benz is credited with the invention of the automobile.*
  - *motorcar* and *automobile* have the same meaning, i.e. they are *synonyms.*
The WordNet Hierarchy

- WordNet synsets correspond to abstract concepts, and they don't always have corresponding words in English.

- These concepts are linked together in a hierarchy. Some concepts are very general, such as *Entity, State, Event* — these are called unique beginners or root synsets.
• *motorcar* has just one possible meaning and it is identified as car.n.01, the first noun sense of *car*.

• The entity car.n.01 is called a **synset**, or "synonym set", a collection of synonymous words (or "lemmas"): 

![WordNet Synset](http://nlds.soe.ucsc.edu)

- **Noun**
  - S: (n) car#1, auto#1, automobile#1, machine#6, motorcar#1 (a motor vehicle with four wheels; usually propelled by an internal combustion engine) "he needs a car to get to work"
Words in Fables

Verb

- [04] S: (n) graze#2 (graze%1:04:00::), grazing#1 (grazing%1:04:01::) (the act of grazing)

- [35] S: (v) crop#5 (crop%2:35:01::), browse#2 (browse%2:35:01::), graze#1 (graze%2:35:01::), range#6 (range%2:35:02::), pasture#2 (pasture%2:35:00::) (feed as in a meadow or pasture) "the herd was grazing"
  - verb group
    - [34] S: (v) range#7 (range%2:34:00::) (let eat) "range the animals in the prairie"
    - [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)
      - direct hypernym / inherited hypernym / sister term
      - derivationally related form
      - sentence frame
    - [35] S: (v) graze#2 (graze%2:35:02::) (break the skin (of a body part) by scraping) "She was grazed by the stray bullet"
    - [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)
    - [35] S: (v) graze#4 (graze%2:35:00::), crease#3 (crease%2:35:02::), rake#6 (rake%2:35:02::) (scrape gently) "graze the skin"
    - [34] S: (v) browse#4 (browse%2:34:00::), graze#5 (graze%2:34:02::) (eat
WordNet (Gloss)

- Each entry has a short definition or **gloss**
- E.g., "A motor vehicle with four wheels; usually propelled by an internal combustion engine"

[WordNet](http://nlds.soe.ucsc.edu)
Words in our Stories

Verb

- [35] S: (v) crop#5 (crop%2:35:01::), browse#2 (browse%2:35:01::), graze#1 (graze%2:35:01::), range#6 (range%2:35:02::), pasture#2 (pasture%2:35:00::) (feed as in a meadow or pasture) "the herd was grazing"
  - verb group
    - [34] S: (v) range#7 (range%2:34:00::) (let eat) "range the animals in the prairie"
    - [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)
      - direct hypernym / inherited hypernym / sister term
      - derivationally related form
      - sentence frame
    - [35] S: (v) graze#2 (graze%2:35:02::) (break the skin (of a body part) by scraping) "She was grazed by the stray bullet"
    - [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)
    - [35] S: (v) graze#4 (graze%2:35:00::), crease#3 (crease%2:35:02::), rake#6 (rake%2:35:02::) (scrape gently) "graze the skin"
    - [34] S: (v) browse#4 (browse%2:34:00::), graze#5 (graze%2:34:02::) (eat..."
WordNet

- Each entry usually has at least one example use from a corpus.
- E.g., "A motor vehicle with four wheels; usually propelled by an internal combustion engine"

Word to search for: motorcar

Display Options: (Select option to change) ▼ Change
Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"
Display options for word: word#sense number

Noun

- S: (n) car#1, auto#1, automobile#1, machine#6, motorcar#1 (a motor vehicle with four wheels; usually propelled by an internal combustion engine) "he needs a car to get to work"

Example
Word Sense Disambiguation

- An area of NLP which focuses on automatically finding the word senses for words in text or dialogue

- Not yet a solved problem

- Classic Example:
  - I took the money to the bank.
  - I went fishing at the bank.

- “Words go in Herds”: Money would rarely co-occur with fishing
Word Sense Disambiguation

Classic Example:

- I took the money to the bank.
- I went fishing at the bank.

WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: bank [Search WordNet]

Display Options: [Select option to change] Change
Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: "an example sentence"

Noun

- S: (n) bank "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"
- S: (n) depository financial institution, bank, banking concern, banking company "he cashed a check at the bank"; "that bank holds the mortgage on my home"
- S: (n) bank "a huge bank of earth"
- S: (n) bank "he operated a bank of switches"
Word Sense: A Sense Key

WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: bank
Search WordNet

Display Options: (Select option to change) ▼ Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: "an example sentence"
Display options for word: word (sense key)

Noun

• S: (n) bank (bank%1:17:01::) "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"
• S: (n) depository financial institution (depository_financial_institution%1:14:00::), bank (bank%1:14:00::), banking concern (banking_concern%1:14:00::), banking company (banking_company%1:14:00::) "he cashed a check at the bank"; "that bank holds the mortgage on my home"
• S: (n) bank (bank%1:17:00::) "a huge bank of earth"
The WordNet Hierarchy

- **S: (n) car#1, auto#1, automobile#1, machine#6, motorcar#1** (a motor vehicle with four wheels; usually propelled by an internal combustion engine) "he needs a car to get to work"
  - *direct hyponym / full hyponym*
    - **S: (n) ambulance#1** (a vehicle that takes people to and from hospitals)
    - **S: (n) funny wagon#1** (an ambulance used to transport patients to a mental hospital)
    - **S: (n) beach wagon#1, station wagon#1, wagon#5, estate car#1, beach waggon#1, station waggon#1, waggon#2** (a car that has a long body and rear door with space behind rear seat)
    - **S: (n) shooting brake#1** (another name for a station wagon)
    - **S: (n) bus#4, jalopy#1, heap#3** (a car that is old and unreliable) "the fenders had fallen off that old bus"
It’s very easy to navigate between concepts. For example, given a concept like *motorcar*, we can look at the concepts that are more specific; the (immediate) hyponyms.
Hypernyms and hyponyms are called **lexical relations** because they relate one synset to another. These two relations navigate up and down the "is-a" hierarchy.
It’s very easy to navigate between concepts. For example, given a concept like *motorcar*, we can look at the concepts that are more specific; the (immediate) **hyponyms**.
We can also navigate up the hierarchy by visiting hypernyms. Some words have multiple paths, because they can be classified in more than one way. There are two paths between `car.n.01` and `entity.n.01` because `wheeled_vehicle.n.01` can be classified as both a vehicle and a container.

- **direct hypernym** / **inherited hypernym** / **sister term**
  - S: (n) motor vehicle#1, automotive vehicle#1
  - S: (n) self-propelled vehicle#1
  - S: (n) wheeled vehicle#1
  - S: (n) vehicle#1
  - S: (n) conveyance#3, transport#1
    - S: (n) instrumentality#3, instrumentation#1
    - S: (n) artifact#1, artefact#1
      - S: (n) whole#2, unit#6
        - S: (n) object#1, physical object#1
          - S: (n) physical entity#1
            - S: (n) entity#1
  - S: (n) container#1
    - S: (n) instrumentality#3, instrumentation#1
      - S: (n) artifact#1, artefact#1
        - S: (n) whole#2, unit#6
          - S: (n) object#1, physical object#1
            - S: (n) physical entity#1
              - S: (n) entity#1
A Lion watched a fat Bull feeding in a meadow, and his mouth watered when he thought of the royal feast he would make, but he did not dare to attack him, for he was afraid of his sharp horns.

There once was a fat bull. The bull was grazing in a meadow. A lion watched the bull and salivated because the lion wanted to eat the bull. The lion didn't attack the bull because the lion feared sharp every horn of the bull.
Words in our Stories

Verb

- [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)

  - direct hypernym / inherited hypernym / sister term
  - derivationally related form
  - sentence frame

- [35] S: (v) graze#2 (graze%2:35:02::) (break the skin (of a body part) by scraping) "She was grazed by the stray bullet"

- [35] S: (v) crop#4 (crop%2:35:10::), graze#3 (graze%2:35:10::), pasture#1 (pasture%2:35:10::) (let feed in a field or pasture or meadow)

- [35] S: (v) graze#4 (graze%2:35:00::), crease#3 (crease%2:35:02::), rake#6 (rake%2:35:02::) (scrape gently) "graze the skin"

- [34] S: (v) browse#4 (browse%2:34:00::), graze#5 (graze%2:34:02::) (eat
Some highly ambiguous Words: Horn

Noun

- (7){03542265} <noun.artifact>[06] S: (n) horn#1 (horn%1:06:06::) (a noisemaker (as at parties or games) that makes a loud noise when you blow through it)
- (3){01328058} <noun.animal>[05] S: (n) horn#2 (horn%1:05:01::) (one of the bony outgrowths on the heads of certain ungulates)
- (1){07280214} <noun.communication>[10] S: (n) horn#3 (horn%1:10:02::) (a noise made by the driver of an automobile to give warning)
- (1){03542111} <noun.artifact>[06] S: (n) horn#4 (horn%1:06:04::), saddle horn#1 (saddle horn%1:06:00::) (a high pommel of a Western saddle (usually metal covered with leather))
- (1){03115320} <noun.artifact>[06] S: (n) cornet#1 (cornet%1:06:00::), horn#5 (horn%1:06:01::), trumpet#1 (trumpet%1:06:00::), trumpet#3 (trumpet%1:06:01::) (a brass musical instrument with a brilliant tone; has a narrow tube and a flared bell and is played by means of valves)
- (1){01328494} <noun.animal>[05] S: (n) horn#6 (horn%1:05:02::) (any hard protuberance from the head of an organism that is similar to or suggestive of a horn)
- {14782206} <noun.substance>[27] S: (n) horn#7 (horn%1:27:00::) (the material (mostly keratin) that covers the horns of ungulates and forms hooves and claws and nails)
- {03542421} <noun.artifact>[06] S: (n) horn#8 (horn%1:06:07::) (a device
A Lion watched a fat Bull feeding in a meadow, and his mouth watered...

There once was a fat bull. The bull was grazing in a meadow.
A Lion watched a fat Bull feeding in a meadow, and his mouth watered...

There once was a fat bull. The bull was grazing in a meadow.
Other Kinds of Links

Word to search for: mellow

Display Options: (Select option to change)

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"

Verb

- **S:** (v) *mellow* (soften, make mellow) "Age and experience mellowed him over the years"
- **S:** (v) *mellow, melt, mellow out* (become more relaxed, easygoing, or genial) "With age, he mellowed"
- **S:** (v) *mellow* (make or grow (more) mellow) "These apples need to mellow a bit more"; "The sun mellowed the fruit"

Adjective

- **S:** (adj) *laid-back, mellow* (unhurried and relaxed) "a mellow conversation"
  - *similar to*
    - **S:** (adj) *relaxed* (without strain or anxiety) "gave the impression of being quite relaxed"; "a relaxed and informal discussion"
  - *derivationally related form*
    - **W:** (n) *mellowness* [Related to: mellow] (geniality, as through the effects of alcohol or marijuana)
  - *antonym*
    - **W:** (adj) *tense* [Indirect via relaxed] (in or of a state of physical or nervous tension)
WordNet: More Lexical Relations

- Some lexical relationships hold between lemmas, e.g., **antonymy**:

  **Noun**
  - S: (n) supply#1
  - S: (n) supply#2
    - direct hypernym / inherited hypernym / sister term
    - **antonym**
      - W: (n) demand#2 [Opposed to: supply]

- There are also relationships between verbs. For example, the act of *walking* involves the act of *stepping*, so walking **entails** stepping. Some verbs have multiple entailments:

  **Verb**
  - S: (v) walk#1 (use one's feet to advance; advance by steps)
    - direct troponym / full troponym
    - verb group
    - direct hypernym / inherited hypernym / sister term
    - entailment
      - S: (v) step#1 (shift or move by taking a step)