LING 408/508: Programming for Linguists

Lecture 26

December 7th
Adminstrivia

• Due Dates
  – Term Programming Project
    • Write-up and code by December 17th
  – Homework 12
    • Install NLTK. Screenshots to show it working. Due Friday 11th
Natural Language Toolkit

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

Natural Language Processing with Python provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more. The book is being updated for Python 3 and NLTK 3. (The original Python 2 version is still available at http://nltk.org/book_ted.)
Installing NLTK

NLTK requires Python versions 2.7 or 3.2+

Mac/Unix

1. Install NLTK: run `sudo pip install -U nltk`
2. Install Numpy (optional): run `sudo pip install -U numpy`
3. Test installation: run `python` then type `import nltk`

For older versions of Python it might be necessary to install setuptools (see [http://pypi.python.org/pypi/setuptools](http://pypi.python.org/pypi/setuptools)) and to install pip (`sudo easy_install pip`).
Install pip

To install pip, securely download `get-pip.py`. [2]

Then run the following (which may require administrator access):

```
python get-pip.py
```

If `setuptools` is not already installed, `get-pip.py` will install `setuptools` for you. [3]

To upgrade an existing `setuptools`, run `pip install -U setuptools`.
PIP (Pip Installs Packages)

```
[dhcp-10-142-184-128:ling508-15 sandiway$ sudo python get-pip.py
[Password: The directory '/Users/sandiway/Library/Caches/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/Users/sandiway/Library/Caches/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting pip
  Downloading pip-7.1.2-py2.py3-none-any.whl (1.1MB) 100% |=================================| 1.1MB 147kB/s
Collecting wheel
  Downloading wheel-0.26.0-py2.py3-none-any.whl (63kB) 100% |=================================| 63kB 1.8MB/s
Installing collected packages: pip, wheel
Successfully installed pip-7.1.2 wheel-0.26.0
dhcp-10-142-184-128:ling508-15 sandiway$
```
Using pip to install nltk

```
[sandwayne@localhost ~]$ which pip
/usr/local/bin/pip

[sandwayne@localhost ~]$ sudo pip install -U nltk
The directory '/Users/sandiway/Library/Caches/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
The directory '/Users/sandiway/Library/Caches/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting nltk
  Downloading nltk-3.1.tar.gz (1.1MB)
    100% 1.1MB 203kB/s
Installing collected packages: nltk
Running setup.py install for nltk
Successfully installed nltk-3.1
```
Install nltk_data

• Command:
  – sudo python -m nltk.downloader -d /usr/local/share/nltk_data all
Test nltk

```python
>>> ^D
dhcp-10-142-184-128:ling508-15 sandiway$ python
Python 2.7.10 (default, Aug 22 2015, 20:33:39)
[GCC 4.2.1 Compatible Apple LLVM 7.0.0 (clang-700.0.59.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> t = nltk.word_tokenize("this is a test.")
>>> t
['this', 'is', 'a', 'test', '.']
>>> tg = nltk.pos_tag(t)
>>> tg
[('this', 'DT'), ('is', 'VBZ'), ('a', 'DT'), ('test', 'NN'), ('.', '.')]  
>>> ne = nltk.chunk.ne_chunk(tg)
>>> ne
Tree('S', [(('this', 'DT'), ('is', 'VBZ'), ('a', 'DT'), ('test', 'NN'), ('.', '.'))])
```
Test nltk_data

```python
>>> from nltk.corpus import brown
>>> brown.words()
[u'The', u'Fulton', u'County', u'Grand', u'Jury', ...]
>>> brown.words()[0]
'u'The'
>>> type(brown.words()[0])
<type 'unicode'>
```
Ubuntu
Ubuntu

setuptools 18.5

*Easily download, build, install, upgrade, and uninstall Python packages*

Package Documentation

Unix including Mac OS X (curl)

If your system has curl installed, follow the `wget` instructions but replace `wget` with `curl`. For example:

```
> curl https://bootstrap.pypa.io/ez_setup.py -o - | python
```
curl

• Command line tool using URL syntax to transfer a file
  - `curl URL` *(downloads file specified in HRL)*
  - can understand http/https and ftp/sftp protocols
setuptools

```
sandiway@sandiway-VirtualBox:~$ curl https://bootstrap.pypa.io/ez_setup.py -o ez_setup.py
| sudo python
[sudo] password for sandiway:  % Total  % Received % Xferd  Average Speed   Time    Time     Current
                                  Dload  Upload   Total  Spent    Left  Speed
100 11432 100 11432  0     0   16336  0     --:--:-- --:--:--    --:--:--

Downloading https://pypi.python.org/packages/source/s/setuptools/setuptools-18.5.zip
Extracting in /tmp/tmppquBAg
Now working in /tmp/tmppquBAg/setuptools-18.5
Installing Setuptools
running install
running bdist_egg
running egg_info
writing requirements to setuptools.egg-info/requires.txt
writing setuptools.egg-info/PKG-INFO
writing top-level names to setuptools.egg-info/top_level.txt
writing dependency_links to setuptools.egg-info/dependency_links.txt
writing entry points to setuptools.egg-info/entry_points.txt
reading manifest file 'setuptools.egg-info/SOURCES.txt'
reading manifest template 'MANIFEST.in'
writing manifest file 'setuptools.egg-info/SOURCES.txt'
installing library code to build/bdist.linux-x86_64/egg
running install_lib
running build_py
creating build
creating build/lib.linux-x86_64-2.7
copying easy_install.py -> build/lib.linux-x86_64-2.7
```
setuptools

byte-compiling build/bdist.linux-x86_64/egg/setuptools/py31compat.py to py31compat.pyc
byte-compiling build/bdist.linux-x86_64/egg/easy_install.py to easy_install.pyc
byte-compiling build/bdist.linux-x86_64/egg/_markerlib/__init__.py to __init__.pyc
byte-compiling build/bdist.linux-x86_64/egg/_markerlib/markers.py to markers.pyc

copying setuptools.egg-info/PKG-INFO -> build/bdist.linux-x86_64/egg/EGG-INF0
copying setuptools.egg-info/SOURCES.txt -> build/bdist.linux-x86_64/egg/EGG-INF0
copying setuptools.egg-info/dependency_links.txt -> build/bdist.linux-x86_64/egg/EGG-INF0
copying setuptools.egg-info/entry_points.txt -> build/bdist.linux-x86_64/egg/EGG-INF0
copying setuptools.egg-info/requires.txt -> build/bdist.linux-x86_64/egg/EGG-INF0
copying setuptools.egg-info/top_level.txt -> build/bdist.linux-x86_64/egg/EGG-INF0

copying setuptools.egg-info/zip-safe -> build/bdist.linux-x86_64/egg/EGG-INF0

creating dist

creating 'dist/setuptools-18.5-py2.7.egg' and adding 'build/bdist.linux-x86_64/egg' to it

removing 'build/bdist.linux-x86_64/egg' (and everything under it)

Processing setuptools-18.5-py2.7.egg

Copying setuptools-18.5-py2.7.egg to /usr/local/lib/python2.7/dist-packages

Adding setuptools 18.5 to easy-install.pth file

Installing easy_install script to /usr/local/bin

Installing easy_install-2.7 script to /usr/local/bin

Installed /usr/local/lib/python2.7/dist-packages/setuptools-18.5-py2.7.egg

Processing dependencies for setuptools==18.5

Finished processing dependencies for setuptools==18.5

sandlway@sandlway-VirtualBox:~$
Ubuntu pip

Easy Install

Easy Install is a python module (easy_install) bundled with setuptools that lets you automatically download, build, install, and manage Python packages.
Ubuntu nltk

```
sandiflower@sandiflower-VirtualBox:~$ which pip
/usr/local/bin/pip
sandiflower@sandiflower-VirtualBox:~$ sudo pip install -U nltk

Collecting nltk
/usr/local/lib/python2.7/dist-packages/pip-7.1.2-py2.7.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:90: InsecurePlatformWarning: A true SSLContext object is not available. This prevents urllib3 from configuring SSL appropriately and may cause certain SSL connections to fail. For more information, see https://urllib3.readthedocs.org/en/latest/security.html#insecureplatformwarning.
  InsecurePlatformWarning
Downloading nltk-3.1.tar.gz (1.1MB)
100% |████████████████████████████████| 1.1MB 148kB/s
Installing collected packages: nltk
Successfully installed nltk-3.1
```

```
sandiflower@sandiflower-VirtualBox:~$
```
Ubuntu nltk_data

```
sandiway@sandiway-VirtualBox:~$ sudo python -m nltk.downloader -d /usr/local/share/nltk_data all
[nltk_data] Downloading collection u'all'
[nltk_data]   | Downloading package abc to
[nltk_data]   |   /usr/local/share/nltk_data...
```
Homework 12

• Install NLTK
  – show it working

1st sentence from the WSJ section of the Penn Treebank

```
dncp-10-142-173-203:Desktop sandiways python
Python 2.7.10 (default, Aug 22 2015, 20:33:39)
[GCC 4.2.1 Compatible Apple LLVM 7.0.0 (clang-700.0.59.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from nltk.corpus import treebank
>>> t = treebank.parsed_sents('wsj_0001.mrg')[0]
>>> t.draw()
```
Homework 12

Pick a sentence of your choosing, tokenize and Part-of-Speech (POS) label it. Comment on whether the labeling is good.

```
>>> import nltk
>>> t = nltk.word_tokenize("'Twas brillig and the slithy toves did gyre and gimble in the wabe")
>>> t
["'T', 'was', 'brillig', 'and', 'the', 'slithy', 'toves', 'did', 'gyre', 'and',
'gimble', 'in', 'the', 'wabe']
>>> t2 = nltk.pos_tag(t)
>>> t2
[([''T'', 'NN'), ('was', 'VBD'), ('brillig', 'VBN'), ('and', 'CC'), ('the', 'DT'),
('slithy', 'JJ'), ('toves', 'NNS'), ('did', 'VBD'), ('gyre', 'NN'), ('and', 'CC'),
'_gimble', 'JJ'), ('in', 'IN'), ('the', 'DT'), ('wabe', 'NN')]`

Tagset documented here
https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html
Homework 12

• Pick another sentence with noun-noun compounds,
  – e.g. Helicopters will patrol the temporary no-fly zone around New Jersey's MetLife Stadium Sunday, with F-16s based in Atlantic City ready to be scrambled if an unauthorized aircraft does enter the restricted airspace.

• Tokenize, POS tag, and extract named entities
Homework 12
Homework 12

GPE = Geo-Political Entity
Natural Language Processing with Python

– Analyzing Text with the Natural Language Toolkit

Steven Bird, Ewan Klein, and Edward Loper

The NLTK book is currently being updated for Python 3 and NLTK 3. This is work in progress; c

0. Preface
1. Language Processing and Python
2. Accessing Text Corpora and Lexical Resources
3. Processing Raw Text
4. Writing Structured Programs
5. Categorizing and Tagging Words (minor fixes still required)
6. Learning to Classify Text
7. Extracting Information from Text
8. Analyzing Sentence Structure
9. Building Feature Based Grammars
10. Analyzing the Meaning of Sentences (minor fixes still required)
11. Managing Linguistic Data (minor fixes still required)
12. Afterword: Facing the Language Challenge

Bibliography
Term Index
Today's Topics

• Chapter 11: Data Collections
  – Lists
  – Arrays
  – Dictionaries (Hash)
Lists

• We've seen these before …
• Example:
  – `range(start,stop,step)`
  – parameters are integers
  – start = 0 by default
  – step = 1 by default

```python
>>> type(range(10))
<type 'list'>
```
Lists

• split:

```python
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> string.split("This is an ex-parrot!")
['This', 'is', 'an', 'ex-parrot!']
```
Lists

• Basic operations:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;seq&gt; + &lt;seq&gt;</code></td>
<td>Concatenation</td>
</tr>
<tr>
<td><code>&lt;seq&gt; * &lt;int-expr&gt;</code></td>
<td>Repetition</td>
</tr>
<tr>
<td><code>&lt;seq&gt;[ ]</code></td>
<td>Indexing</td>
</tr>
<tr>
<td><code>len(&lt;seq&gt;)</code></td>
<td>Length</td>
</tr>
<tr>
<td><code>&lt;seq&gt;[: ]</code></td>
<td>Slicing</td>
</tr>
<tr>
<td><code>for &lt;var&gt; in &lt;seq&gt;</code></td>
<td>Iteration</td>
</tr>
<tr>
<td><code>&lt;expr&gt; in &lt;seq&gt;</code></td>
<td>Membership check (Returns a Boolean)</td>
</tr>
</tbody>
</table>

```python
>>> a = [1,2,3]
>>> b = [4,5,6]
>>> a + b
[1, 2, 3, 4, 5, 6]
>>> a * 5
[1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3]
>>> a * b
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can't multiply sequence by non-int of type 'list'
```
Lists

```python
>>> a = [1, 2, 3]
>>> a * (5 / 3)
[1, 2, 3]
>>> a * (7 / 3)
[1, 2, 3, 1, 2, 3]
>>> a * (7 / 3.0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can't multiply sequence by non-int of type 'float'
```
Lists

```python
>>> a[1]  # a = [1,2,3]
2
>>> a[0]
1
>>> a[2]
3
>>> a[3]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

```python
>>> a = [1,2,3,4,5]
>>> a[1:3]
[2, 3]
>>> a[1:]
[2, 3, 4, 5]
>>> a[:2]
[1, 2]
>>> a[:]
[1, 2, 3, 4, 5]
```

```python
>>> a[1:3]
[2, 3]
>>> a[1:]
[2, 3, 4, 5]
>>> a[:2]
[1, 2]
>>> a[:]
[1, 2, 3, 4, 5]
>>> 2 in a
True
>>> 7 in a
False
>>> c = 5
>>> c in a
True
>>> c in b
True
```
Lists

• Lists are malleable:

<table>
<thead>
<tr>
<th>Method</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;list&gt;.append(x)</td>
<td>Add element x to end of list.</td>
</tr>
<tr>
<td>&lt;list&gt;.sort()</td>
<td>Sort (order) the list. A comparison function may be passed as parameter.</td>
</tr>
<tr>
<td>&lt;list&gt;.reverse()</td>
<td>Reverse the list.</td>
</tr>
<tr>
<td>&lt;list&gt;.index(x)</td>
<td>Returns index of first occurrence of x.</td>
</tr>
<tr>
<td>&lt;list&gt;.insert(i,x)</td>
<td>Insert x into list at index i.</td>
</tr>
<tr>
<td>&lt;list&gt;.count(x)</td>
<td>Returns the number of occurrences of x in list.</td>
</tr>
<tr>
<td>&lt;list&gt;.remove(x)</td>
<td>Deletes the first occurrence of x in list.</td>
</tr>
<tr>
<td>&lt;list&gt;.pop(i)</td>
<td>Deletes the ith element of the list and returns its value.</td>
</tr>
</tbody>
</table>

Can implement stacks and queues easily
Lists

```
list.append(x)
    Add an item to the end of the list; equivalent to a[len(a):] = [x].

list.extend(L)
    Extend the list by appending all the items in the given list; equivalent to a[len(a):] = L.

list.insert(i, x)
    Insert an item at a given position. The first argument is the index of the element before which to insert, so a.insert(0, x) inserts at the front of the list, and a.insert(len(a), x) is equivalent to a.append(x).

list.remove(x)
    Remove the first item from the list whose value is x. It is an error if there is no such item.

list.pop(i)
    Remove the item at the given position in the list, and return it. If no index is specified, a.pop() removes and returns the last item in the list. (The square brackets around the i in the method signature denote that the parameter is optional, not that you should type square brackets at that position. You will see this notation frequently in the Python Library Reference.)

list.index(x)
    Return the index in the list of the first item whose value is x. It is an error if there is no such item.

list.count(x)
    Return the number of times x appears in the list.

list.sort(cmp=None, key=None, reverse=False)
    Sort the items of the list in place (the arguments can be used for sort customization, see sorted() for their explanation).

list.reverse()
    Reverse the elements of the list, in place.
```
Lists

```python
>>> a
[1, 2, 3, 4, 5]
>>> a.append(6)
[1, 2, 3, 4, 5, 6]

>>> a = [3,1,4,1,5]
>>> b = a.sort()
>>> a
[1, 3, 3, 4, 5]
>>> a.reverse()
>>> a
[5, 4, 3, 1, 1]

>>> s = ['the','name','of','this','list']
>>> s.sort()
>>> s
['list', 'name', 'of', 'the', 'this']
>>> a = [3,1,4,1,5]
>>> a.sort()
>>> a
[1, 1, 3, 4, 5]
>>> b = a
>>> a.sort()
>>> a
[1, 1, 3, 4, 5]
>>> b
[1, 1, 3, 4, 5]
```
Lists

>>> a
[5, 4, 3, 1, 1]
>>> a.index(1)
3
>>> a.insert(2,100)
>>> a
[5, 4, 100, 3, 1, 1]
>>> a.insert(len(a),6)
>>> a
[5, 4, 100, 3, 1, 1, 6]
>>> a.count(1)
2
>>> a.count(100)
1
>>> a.remove(1)
>>> a
[7, 5, 4, 100, 3, 1, 6]

>>> a.remove(99)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: list.remove(x): x not in list
>>> a.pop(2)
4
>>> a
[7, 5, 100, 3, 1, 6]
>>> a.pop(len(a))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: pop index out of range
>>> a.pop(len(a)-1)
6
>>> a
[7, 5, 100, 3, 1]
Lists

>>> a = []
>>> a
[]
>>> a.append(1)
>>> a
[1]
>>> a.append(2)
>>> a
[1, 2]
>>> a.pop(len(a)-1)
2
>>> a
[1]
>>> a.insert(0,2)
>>> a
[2, 1]
>>> a.pop(0)
2
>>> a
[1]

Queue #1: list.append(x) and list.pop(0)

>>> a = []
>>> a.append(1)
>>> a.append(2)
>>> a.append(3)
>>> a
[1, 2, 3]
>>> a.pop(0)
1
>>> a.pop(0)
2
>>> a.pop(0)
3
>>> a
[]
Lists

Queue #2: `list.insert(0,x)` and `list.pop(len(list)-1)`

```
>>> a
[]
>>> a.insert(0,1)
>>> a.insert(0,2)
>>> a.insert(0,3)
>>> a
[3, 2, 1]
>>> a.pop(len(a)-1)
1
>>> a.pop(len(a)-1)
2
>>> a.pop(len(a)-1)
3
```

non-object-oriented function: `del`

```
>>> a = [3,1,4,1,5]
>>> del a[2]
>>> a
[3, 1, 1, 5]
>>> del a[1:3]
>>> a
[3, 5]
```
Higher order functions

map(function,list)
reduce(function,list)

inline anonymous function: lambda
example: \( \lambda x. x+x \)    lambda \( x: x+x \)

```
>>> map(lambda x: x+x, [1,2,3])
[2, 4, 6]

>>> reduce(lambda x,y: x+y, [1,2,3])
6

>>> a
[3, 5]

>>> a = [3,1,4,5,6,2]

>>> reduce(lambda x,y: x+y, a) / len(a)
3

>>> reduce(lambda x,y: x+y, a) + 0.0 / len(a)
3.5

filter(function,list)

```
Standard Deviation Example

```python
import math
def sum(x,y):
    return x+y

def mean(nums):
    return (reduce(lambda x,y:x+y,nums) + 0.0)/len(nums)

def stdDev(nums):
    m = mean(nums)
    sumsqs = reduce(lambda x,y:x+(y-m)*(y-m),nums)
    return math.sqrt(sumsqs/(len(nums)-1))

def sDev(nums):
    m = mean(nums)
    sum = 0.0
    for num in nums:
        dev = m - num
        sum += dev * dev
    return math.sqrt(sum/(len(nums)-1))
```