HW#1 & Getting Started

LING 571 — Deep Processing Techniques for NLP
September 28, 2022
Shane Steinert-Threlkeld

Department Cluster

- Assignments are **required** to run on department cluster
  - If you don’t have a cluster account, request one ASAP!
    - Link to account request form on Canvas or below:
    - https://cldb.ling.washington.edu/live/accountrequest-form.php
- You are not required to develop on the cluster, but code must run on it

Department Cluster

- Assignments are **required** to run on department cluster
  - If you don’t have a cluster account, request one ASAP!
    - Link to account request form on Canvas or below:
    - https://cldb.ling.washington.edu/live/accountrequest-form.php
- You are not required to develop on the cluster, but code must run on it
- ***Reminder:*** *All but most simple tasks must be run via Condor*

Condor

- Parallel computing management system
- All homework will be run via condor
- See documentation on CLMS wiki for:
  - Construction of condor scripts
  - Link also on course page under “Course Resources”

NLTK

- Most assignments will use NLTK in Python
- **N**atural **L**anguage **T**ool**K**it (NLTK)
  - Large, integrated, fairly comprehensive
    - Stemmers
    - Taggers
    - Parsers
    - Semantic analysis
    - Corpus samples
    - …& More
  - Extensively documented
  - Pedagogically Oriented
    - Implementations Strive for Clarity
    - …sometimes at the expense of efficiency.

NLTK

- nltk.org
  - Online book
  - Demos of software
  - How-Tos for specific components
  - API information, etc.

Python & NLTK

- NLTK is installed on the Cluster
  - Use Python 3.4+ with NLTK
  - **N.B.:** Python 2.7 is default
    - More/later versions in `/opt/python-*/bin/`
    - You can make a personal alias, but your bash scripts will not run in your personal environment, so keep that in mind.
    - **Use full paths to python binary in your scripts to run!**
      - e.g. /opt/python-3.6/bin/python3 your_file.py
- Data is also installed:
  - /corpora/nltk/nltk-data
- Written in Python
  - Some introductions at: **python.org**, **docs.python.org**
  - Orientation tutorial: https://github.com/shanest/python-tutorial-clms

Python & NLTK

- Interactive mode allows experimentation, introspection:

```
patas$ python3
>>> import nltk
>>> dir(nltk)
['AbstractLazySequence', 'AffixTagger', 'AlignedSent',
'Alignment', 'AnnotationTask', 'ApplicationExpression',
'Assignment', 'BigramAssocMeasures', 'BigramCollocationFinder',
'BigramTagger', ‘BinaryMaxentFeatureEncoding’,…
>>> help(nltk.AffixTagger)
```

Turning In Homework

- Will be using Canvas’ file submission mechanism
  - Quick how to at: https://community.canvaslms.com/docs/DOC-10663-421254353

Turning In Homework

- Will be using Canvas’ file submission mechanism
  - Quick how to at: https://community.canvaslms.com/docs/DOC-10663-421254353
- Homeworks due on **Wednesday** nights

Turning In Homework

- Will be using Canvas' file submission mechanism
  - Quick how to at: https://community.canvaslms.com/docs/DOC-10663-421254353
- Homeworks due on **Wednesday** nights
- 11:59 PM, Pacific Time

Turning In Homework

- Will be using Canvas' file submission mechanism
  - Quick how to at: https://community.canvaslms.com/docs/DOC-10663-421254353
- Homeworks due on **Wednesday** nights
- 11:59 PM, Pacific Time
- Generally, each assignment will include:
  - `readme.{txt|pdf}`
  - `hwX.tar.gz`
    - Where "X" is the assignment number
    - **`tar -cvzf hwX.tar.gz <hw_path>`**

HW #1

- Read in sentences and corresponding grammar
- Use NLTK to parse those sentences
- Goals:
  - Set up software environment for rest of course
  - Get familiar with NLTK
  - Work with parsers and CFGs

HW #1: Useful Tools

- Loading data:
  - **`nltk.data.load(resource_url)`**
    - Reads in and processes formatted CFG/FCFG/treebank/etc
    - Returns a grammar from CFG
    - **examples:**

```
nltk.data.load(‘grammars/sample_grammars/toy.cfg’)
nltk.data.load(‘file://' + my_grammar_path)
```

    - (NB: absolute path!)

HW #1: Useful Tools

- Loading data:
  - **`nltk.data.load(resource_url)`**
    - Reads in and processes formatted CFG/FCFG/treebank/etc
    - Returns a grammar from CFG
    - **examples:**

```
nltk.data.load(‘grammars/sample_grammars/toy.cfg’)
nltk.data.load(‘file://' + my_grammar_path)
```

    - (NB: absolute path!)

- Tokenization:
  - **`nltk.word_tokenize(mystring)`**
    - Returns array of tokens in string

HW #1: Useful Tools

- Parsing:
  - **`parser = nltk.parse.EarleyChartParser(grammar)`**
    - Returns parser based on the grammar
  - **`parser.parse(token_list)`**
    - Returns iterator of parses:

```
>>> for item in parser.parse(tokens):
>>>     print(item)

(S (NP (Det the) (N dog)) (VP (V chased) (NP (Det the) (N cat))))
```