Introduction to Human Language Technologies
Lab.5: Lexical Semantics

Natural Language Research Group

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Course 2018/19

Outline

Session requirements

Polarity:

- Both Linux & Windows (via python shell)

```
> import nltk
> nltk.download('movie_reviews')
```

Outline

Examples

wordnet reader
(`http://www.nltk.org/howto/wordnet.html`)

Provide an interface to access WordNet data, such as:

- synsets of a given lemma+PoS pair,
- lemmas of a given synset,
- hypernyms and hyponyms of a given synset,
- synonyms and antonyms of a given lemma in a synset
- least common subsumers of a pair of synsets
- different measures of synset similarity

...

Examples (I)


```
In [1]: from nltk.corpus import wordnet as wn
```

synsets

```
In [2]: wn.synsets('age','n')
Out[2]: [Synset('age.n.01'),
         Synset('historic_period.n.01'),
         Synset('age.n.03'),
         Synset('long_time.n.01'),
         Synset('old_age.n.01')]
```

```
In [3]: age = wn.synset('age.n.1')
        age
Out[3]: Synset('age.n.01')
```

Examples (II)

definitions, examples and lemmas

```
In [4]: age.definition()
Out[4]: 'how long something has existed'

In [5]: age.examples()
Out[5]: ['it was replaced because of its age']

In [6]: ls = wn.synsets('age','n')
        ll = ls[1].lemmas()
        [lemma.name() for lemma in ll]
Out[6]: ['historic_period', 'age']
```

antonyms

```
In [11]: good = wn.synset('good.a.01')
         good.lemmas()[0].antonyms()
Out[11]: [Lemma('bad.a.01.bad')]
```

Examples (III)

hyponyms and hypernyms

```
In [7]: age.hyponyms()
```

```
Out[7]: [Synset('bone_age.n.01'),
         Synset('chronological_age.n.01'),
         Synset('developmental_age.n.01'),
         Synset('fetal_age.n.01'),
         Synset('mental_age.n.01'),
         Synset('newness.n.01'),
         Synset('oldness.n.01'),
         Synset('oldness.n.02'),
         Synset('youngness.n.01')]
```

```
In [8]: age.hypernyms()
```

```
Out[8]: [Synset('property.n.02')]
```

```
In [9]: age.root_hypernyms()
```

```
Out[9]: [Synset('entity.n.01')]
```

Examples (IV)


```
In [9]: hyper = lambda s: s.hypernyms()
        list(age.closure(hyper))

Out[9]: [Synset('property.n.02'),
         Synset('attribute.n.02'),
         Synset('abstraction.n.06'),
         Synset('entity.n.01')]

In [10]: age.tree(hyper)

Out[10]: [Synset('age.n.01'),
          [Synset('property.n.02'),
           [Synset('attribute.n.02'),
            [Synset('abstraction.n.06'), [Synset('entity.n.01')]]]]]
```

Examples (V)

all lexical relations

In [23]:

```
def getRelValue(name):
    method = getattr(age, rel)
    return method()

lexRels = ['hypernyms', 'instance_hypernyms', 'hyponyms', 'instance_hyponyms', \
           'member_holonyms', 'substance_holonyms', 'part_holonyms', \
           'member_meronyms', 'substance_meronyms', 'part_meronyms', \
           'attributes', 'entailments', 'causes', 'also_sees', 'verb_groups', 'similar_tos']
age = wn.synset('age.n.01')

results = {}
for rel in lexRels:
    val = getRelValue(rel)
    if val != []:
        results[rel] = val
results
```

Out[23]:

```
{'attributes': [Synset('immature.a.04'),
  Synset('mature.a.03'),
  Synset('new.a.01'),
  Synset('old.a.01'),
  Synset('old.a.02'),
  Synset('young.a.01')],
 'hypernyms': [Synset('property.n.02')],
 'hyponyms': [Synset('bone_age.n.01'),
  Synset('chronological_age.n.01'),
  Synset('developmental_age.n.01'),
  Synset('fetal_age.n.01'),
  Synset('mental_age.n.01'),
  Synset('newness.n.01'),
  Synset('oldness.n.01'),
  Synset('oldness.n.02'),
  Synset('youngness.n.01')]}
```

Optional exercise

Statement:

1. Develop a function to search and show the shortest path between two noun synset.
2. Apply it to show the shortest path between *dog.n.01* and *cat.n.01*.

Note:

- A good way to show the results is like a tree.

Outline

Example

similarities

In [12]:
```
dog = wn.synset('dog.n.01')
cat = wn.synset('cat.n.01')
dog.path_similarity(cat)
```

Out[12]: 0.2

In [13]:
```
dog.lch similarity(cat)
```

Out[13]: 2.0281482472922856

In [14]:
```
dog.wup_similarity(cat)
```

Out[14]: 0.8571428571428571

In [15]:
```
from nltk.corpus import wordnet_ic
brown_ic = wordnet_ic.ic('ic-brown.dat')
dog.lin_similarity(cat,brown_ic)
```

Out[15]: 0.8768009843733973

In [16]:
```
dog.lowest common hypernyms(cat)
```

Out[16]: [Synset('carnivore.n.01')]

Mandatory exercise

Given the following (lemma, category) pairs:

('the','DT'), ('man','NN'), ('swim','VB'), ('with', 'PR'), ('a', 'DT'), ('girl','NN'), ('and', 'CC'), ('a', 'DT'), ('boy', 'NN'), ('whilst', 'PR'), ('the', 'DT'), ('woman', 'NN'), ('walk', 'VB')

For each pair, when possible, print their most frequent WordNet synset, their corresponding least common subsumer (LCS) and their similarity value, using the following functions:

- Path Similarity
- Leacock-Chodorow Similarity
- Wu-Palmer Similarity
- Lin Similarity

Normalize similarity values when necessary. What similarity seems better?

Outline

NLTK's Movie Reviews

- polarity corpus
  - 1000 positive examples
  - 1000 negative examples
- use in NLTK

```
In [20]:
1 from nltk.corpus import movie_reviews as mr
2
3 mr.fileids('pos')[:5]
```

```
Out[20]: ['pos/cv000_29590.txt',
          'pos/cv001_18431.txt',
          'pos/cv002_15918.txt',
          'pos/cv003_11664.txt',
          'pos/cv004_11636.txt']
```

```
In [23]:
1 mr.words('pos/cv000_29590.txt')[:7]
```

```
Out[23]: ['films', 'adapted', 'from', 'comic', 'books', 'have', 'had']
```

SentiWordnet Example

SentiWordnet in NLTK:

SentiWordnet

In [10]:
```
# getting the wordnet synset
synset = wn.synset('good.a.1')
# gettin the sentiwordnet synset
sentiSynset = swn.senti_synset(synset.name())
# getting the scores: positivity, negativity and objectivity
sentiSynset.pos_score(), sentiSynset.neg_score(), sentiSynset.obj_score()
```

Out[10]: (0.75, 0.0, 0.25)

Optional exercise

Statement (unsupervised polarity system):

1. Get the first synset (most frequent) of one of the next alternatives:
   - nouns, verbs, adjectives and adverbs
   - nouns, adjectives and adverbs
   - only adjectives
2. Sum all the positive scores and negative ones to get the polarity
3. Apply the system to the *movie reviews* corpus and give the accuracy
4. Give some conclusions about the work

Notes:

- We can assign the proper sense, instead of the first one, using a Word Sense Disambiguation tagger. We will see them next session.