Project Text Analysis – Assignment 2

Malvina Nissim, Lennart Kloppenburg
m.nissim@rug.nl, lennyklb@gmail.com

29 April 2015

The aim of this assignment is to get you acquainted with classifying entities. Thus, you will use Wordnet for giving a class to entities represented by common nouns, and a Named Entity Recogniser to classify Named Entities. Remember that in order to do both tasks, you will have to have reached already some stage in the text processing pipeline, so you will have to reuse some knowledge from last assignment, as your text will have to be processed up to POS tags. Also, please remember that you find a lot of information (and code) on the slides from Lecture 3, so you can refer to those, too (on Nestor).

- use Nestor for submission
- **Deadline**: Wednesday 6 May 2015, 23:59
- hand in three files, two `.py` and one `.txt`/`.pdf`:
  - one python script that generates all the information required in Exercise 1
  - one python script that generates all the information required in Exercise 2
  - a `.txt` or `.pdf` file (please no `.docx`) with all comments/answers to the questions in both exercises. Please, use two sections in your document, one for Exercise 1 and one for Exercise 2. If your scripts generate (almost) all the information you need in a nice informative way you can just copy and paste it from your code's output, of course. Add comments where asked by the exercise and where appropriate.

Exercise 1 – WordNet

hint

for this exercise you can find helpful information here:
`http://www.nltk.org/howto/wordnet.html`
including all similarity measures implemented in the WordNet module and how to use them.

1. WordNet relations.

   From the text file provided (the Wikipedia article about Ada Lovelace), exploiting WordNet, you will have to find the following information:

   a) how many nouns refer to a **relative**? (excluding names)

   b) how many nouns refer to an **illness**?

   c) how many nouns refer to a **science**?

   In each case: which words?

   Helpful code (see also slides)

```python
def hypernymOf(synset1, synset2):
    """ Returns True if synset2 is a hypernym of
    synset1, or if they are the same synset.
    Returns False otherwise. """
    if synset1 == synset2:
        return True
    for hypernym in synset1.hypernyms():
        if synset2 == hypernym:
            return True
        if hypernymOf(hypernym, synset2):
            return True
    return False
```

   Helpful steps

   - Create a list of the lemmas of all nouns, `noun_lemmas` (each occurrence, not unique nouns). Note that you can simply expand/modify the code in the slides, for this. The example there is verbs, but you can use it for nouns, of course.
   - Retrieve the synsets for each lemma.
   - Check whether it's a hyponym of one of the three synsets of interest.
   - Count the relevant nouns, and collect them.

2. Multiple classes.

   As we have seen, one word can belong to more than one synset and thus have more than one possible hypernym. Classify every noun which occurs in your text into the 25 top noun classes in WordNet (see slides). Then answer these questions:

   - in how many cases was there only one hypernym per noun? Give a couple of examples, indicating the noun and its hypernym
   - in how many cases did your system had to choose among more than one possible hypernyms? Give a couple of examples, and specify which hypernyms were available
   - what is the average number of hypernyms per noun in the whole text?

3. WordNet similarity.[1]

   In their article "Contextual correlates of semantic similarity", published in the journal *Language and Cognitive Processes* in 1991, the psychologists George Miller and Walter Charles report the results of an experiment where subjects were asked to judge how similar the meaning of a pair of words was on a 5-point scale from 0 to 4. These are the mean similarity scores obtained for 6 noun pairs (out of a total of 30 pairs), with two pairs representing high-level similarity, two pairs representing medium-level similarity, and two pairs representing low-level similarity:

   ```
   car-automobile 3.92
   coast-shore 3.70
   food-fruit 3.08
   journey-car 1.16
   monk-slave 0.55
   moon-string 0.08
   ```

   Use one or more of the predefined similarity measures in the NLTK corpus reader for WordNet to score the similarity of each of these word pairs. Rank the pairs in order of decreasing similarity and discuss how close your ranking is to the one obtained experimentally by Miller and Charles. (Note that what matters is the ranking, not the actual scores). If you want to read the original article, or just see the whole list of pairs, you can find it here: `http://www.tandfonline.com/doi/pdf/10.1080/01690969108406936`.

Exercise 2 – Named Entity Recognition

hint

you can find information about topics in this portion of the exercise here:
`http://www.nltk.org/book/ch07.html`

For this exercise you will run one of the best Named Entity Recognisers available, namely the Stanford NE Tagger. NLTK provides a nice interface to the Stanford tools, but you will have to read some documentation and download some software first:

- Read about the Stanford Named Entity Recognizer here:
  `http://nlp.stanford.edu/software/CRF-NER.shtml`
- Download software:
  `http://nlp.stanford.edu/software/stanford-ner-2014-06-16.zip`

[1]Thanks to Raquel Fernandez (UvA) for this exercise.

Afterwards, you can import it and use it like this:

```
>>> from nltk.tag.stanford import NERTagger
>>> st = NERTagger('stanford-ner-2014-06-16/classifiers/english.all.3class.distsim.crf.ser.gz',
...'stanford-ner-2014-06-16/stanford-ner-3.4.jar')
>>> st.tag('Frank de Boer studies at the University of Groningen in The Netherlands'.split())
[('Frank', 'PERSON'), ('de', 'PERSON'), ('Boer', 'PERSON'), ('studies', 'O'), ('at', 'O'),
...('the', 'O'), ('University', 'ORGANIZATION'), ('of', 'ORGANIZATION'),
...('Groningen', 'ORGANIZATION'), ('in', 'O'), ('The', 'LOCATION'), ('Netherlands', 'LOCATION')]
>>>
```

What you have to do:

1. run the Named Entity Recognizer on your file, and count how many Persons, Organizations, and Locations you find. Have a look at them - are they all correct?

2. as you can see there are only three Named Entity classes. Have a look at the other models that are contained in the directory of the stanford tagger that you have downloaded and then used in `nltk`, and combining this information with what you find at the link provided above, experiment with other classification sets. Please, comment on what the other classifiers do, and on what happens.

3. pick the model you prefer, and using **both** the NER tagger and WordNet information, classify all nouns or NP chunks in the text using the classes expressed by your classifier (for example, picking the classifier from the box above, this would mean classifying all nouns or NP chunks into one of Person, Organization, Location.)