Homework 9:
WordNet

Benjamin Roth, Marina Sedinkina
Symbolische Programmiersprache

Due: Thursday January 18, 2017, 16:00

In this exercise you will:

- measure semantic similarity of words using WordNet
- calculate WordNet polysemy
- compute lesk similarity of two concepts using WordNet

This homework will be graded using unit tests by running: `python3 -m unittest -v hw09_wordnet/test_wordnet.py`

Exercise 1: WordNet semantic similarity [1 point]

Use the predefined path-based similarity measures (accessible with the use of `synset1.path_similarity(synset2)`) to score the similarity of each of the following pairs of words: car-automobile, gem-jewel, journey-voyage, boy-lad, coast-shore, asylum-madhouse, magician-wizard, midday-noon, furnace- stove, food-fruit, bird-cock, bird-crane, tool-implement, brother-monk, lad- brother, crane-implement, journey-car, monk-oracle, cemetery-woodland, food- rooster, coast-hill, forest-graveyard, shore-woodland, monk-slave, coast-forest, lad-wizard, chord-smile, glass-magician, rooster-voyage, noon-string.

- In `noun_similarity.py` implement the function `get_similarity_scores(pairs)` so that it ranks the pairs in order of decreasing similarity. **Hint**: the similarity of a pair should be represented by the similarity of the most similar pair of synsets they have.

Exercise 2: WordNet polysemy [1 point]

The polysemy of a word is the number of senses it has. Using WordNet, we can determine that the noun dog has 7 senses.

- In `average_polysemy.py` implement the function `average_polysemy(part_of_speech)` so that it computes the average polysemy of nouns, verbs, adjectives and adverbs according to WordNet. Hint: look up the documentation for the method `nltk.corpus.wordnet.all_lemma_names()`, which you can use to get the synsets for a given POS.
- run the code with `python3 -m hw09_wordnet.average_polysemy`

Exercise 3: Lesk similarity [4 points]

Lesk similarity is defined as the textual overlap between the corresponding definitions, as provided by a dictionary. Take a look at `hw09_wordnet/lesk_sim.py`. In this exercise you will have to implement some methods to measure lesk similarity of two concepts.

- Complete the class method `get_definition_words(self,synset)`. This method should find tokens of wordnet definition of synset. Punctuation in definitions should be eliminated, because they do not have a meaning. If two definitions contain punctuation, the score increases.
- Complete the class method

  `get_max_match(self,definition_words1,definition_words2)`. This method should calculate maximum matching number (the length of shortest definition). The larger a text, the higher can its score be. Maximum matching number will be used for normalization to allow a fair comparison.
- Complete the class method

  `get_overlap(self,definition_words1,definition_words2)`. This method should find overlap in definitions considering words occuring twice.
- Complete the class method `score(self)`. This method should return lesk similarity. Use methods defined in the class to compute it. **Hint**: use formula `overlap/max_match`.