CSC 447 Programming Assignment #2 **Due Mar. 21th**

The goal of this assignment is to compare a range of different techniques to judge word similarity and test them on a task where you are given three words as input, say "dog cat house" and you return the two words that are closest, in this case "dog" and "cat", plus the similarity score for this pair.

You will be building three standard measures and then will develop your own approach for a fourth. The standard three will be

1) the Word2vec metric (see details below for obtaining this code)
2) a Wu-Palmer metric on the TRIPS ontology (see details below on getting the packages to access TRIPS)
3) a word-vector approach that you will compute off the Brown corpus file we provide.
4) You can choose anything you want for the **fourth technique**, which could involve combining the above measures, or training better vectors using more corpora, or whatever else you can think of. The point is to build some representation yourself. Using a package that trains a model for you would not be acceptable.

In order to simplify the assignment, we will only be considering the nouns and verbs that are actually defined in the TRIPS lexicon and occur in the Brown corpus - about 4800 words. The list of words is attached. We are also providing a lemmatized version of the Brown Corpus that is also all lowercase to make things simpler.

Here's the information on the resources you need.

Word2vec

To load pre-trained Word2vec vectors, read link. To use a built-in similarity function, see their 'genism.models.keyedvectors'. In your documentation, please specify which model you use.

TRIPS lexicon

The lex-ont.json file contains a mapping from a word in the lexicon to its TRIPS ontology type (labeled as "lf_parent")

TRIPS Ontology

To get access to the TRIPS ontology, first install jsontrips which provides the TRIPS ontology as a JSON file for easy loading.

pip3 install jsontrips

To load jsontrips, type

```
import jsontrips
ontology_dict = jsontrips.ontology()
```

Then, ontology_dict should return a dictionary containing information about 3758 TRIPS ontology types. Let's look at the ontology type "MOVE". Note all type names are capitalized.

```
ontology_dict['MOVE'].keys()
```

should return:

dict_keys(['arguments', 'children', 'name', 'parent', 'sem', 'wordnet_sense_keys'])

To measure Wu-Palmer similarity, you'd want to use 'children' and 'parent' keys.

For example, jsontrips.ontology()["MOVE"]['children'] should return **the list of children of the type "MOVE" in the ontology**.

And jsontrips.ontology()["MOVE"]['parent'] should return **the parent of "MOVE", "MOTION".**

Testing

We will provide a small development test set, which provides a set of triples as input as described below. Your final program should read each triple and output a list of answers,

1) the first being the results of Word2vec,
2) the second the results Wu-Palmer on TRIPS,
3) the third your word vector approach,
4) and the last your "novel" technique.

For each approach you should identify the two words that are most similar, and the similarity score your program attained.

We will provide a set of gold answers for the development set.

Details for Submission of your Program

Submit your program and the documentation on Blackboard. Your program should be defined so that it can be run with the command line

python3 [yourprogram].py [input-file] [output-file]

and input file we will provide would look like this:

"dog cat house"

"bottle house run"

and your output should be like this (where some of these numbers are made up!):

"dog cat .76094574 dog cat 1.0 dog house .5 dog cat 0.8"

"bottle house 0.19791234 house run 0.8 bottle house .5 bottle run 0.4"

We will be testing your program on some additional unseen data.

NLP_prog2_Write up_Jingwen Zhong

In your documentation, you should compare the results from the three approaches and your novel approach to the gold and discuss why you think each approach worked well or worked poorly. You also should document your novel approach and the rationale behind it.

Elaborations

- Note that for the **TRIPS approach** you have to anticipate that words might be ambiguous, and have multiple senses in the lexicon. You should choose a reasonable strategy – say finding the two closest senses and reporting that. You should describe your approach in your documentation.

- Given that the approaches are prescribed in this assignment except for the fourth approach, you will not be graded on how good the results are for each approach. But you will be grade on whether you computed **the scores correctly**. Also, we'll assign a few bonus points for exceptional performance of your novel technique.

- With only a million words, the **Brown corpus** is too small to expect good results from the computed vectors, but it will be interesting to see how it performs. You should pre-process the file to remove all special characters, so the context of a word only includes letters and numbers. When building your vectors for **approach 3**, you should limit the words in the vectors to the words on the provided list and use a context window of **plus or minus 4 words**. For example, say the provided lexicon has words A, B, C, D, and E, and one of the occurrences of A in Brown is in this context

  X Y B S D A B R T C E

  You would add 2 B's, 1 C and 1 D to the vector for A. Y, S, and T are not included as they are not in the lexicon, and the E at the end is not included as it is outside the window.

  Your custom approach could experiment using expanded vectors or be a combination of the results of the first three systems.

- Finally, also note that the testing has some bias towards the **TRIPS ontology** approach, as we are only selecting words that are defined in the lexicon, as opposed allowing arbitrary words. We could allow any words by using the **TRIPS-WordNet ontology** but wanted to avoid the extra complexity of including the WordNet information.