SHW5: WordNet

November 6, 2018; due November 13, 2018 (11:59pm)

In this homework, you will be exploring WordNet by identifying hyponyms of synset and finding them in texts, as well as building clusters from synset to match words that are out of the WordNet vocabulary. The iPython notebook to be completed and necessary data can be found on bCourses in `Files/SHW5/`. Turn in your homework on Gradescope to be evaluated.

You are given the following files:

- `SHW5.ipynb` : This homework's iPython notebook
- `index.noun` : WordNet file relating nouns to their synsets
- `data.noun` : WordNet file relating noun synsets to their hyponyms
- `literary.texts.txt` : Example text to find synsets from
- `glove.twitter.27B.25d.txt` : GloVe embeddings trained from twitter

Please do **not** add any additional files or import any additional libraries beyond the ones given for this homework.

1 Hyponym Identification

For this problem, you will be using the WordNet hyponym tree in order to identify all occurences of a hyponym of a given synset in a piece of text. Functions to parse through the `index.noun` and `data.noun` files are provided, and will give you mappings between words and synset IDs as well as a hyponym tree, represented as a mapping between a synset and its direct hyponyms. This problem is done in two parts:

1.1 The first thing that you would need to do is to implement `get_hyponym_terms`, which returns all the hyponyms of a given synset. To do so, you'll navigate through the tree starting at the given synset, and collecting all of the descendents of that synset in the tree. At the end of the function, you should return a set of all the hyponym terms (not the synset IDs!) of the given synset.

1.2 Next, implement `get_synset_locations`, which, given a word, finds and records the location of all instances and any hyponym of that word. Details about how each location should be recorded can be found in the iPython notebook.

2 Synset Clustering

For this problem, you will be generating clusters from synsets in order to find the synset most similar to any arbitrary word, as long as that word is a word embedding. We will do this by using GloVe embeddings, trained from Twitter, and build clusters by finding representative vectors for each synset.

2.1 Implement `create_clusters`, which returns a dictionary containing a mapping between every synset and the representative vector for that synset. The representative vector should maximize the cosine similarity of the embeddings of all the words in the synset, or in other words, it is the vector that is closest to all the words in the synset when measured by cosine similarity. Keep in mind that when measuring cosine similarity, vectors are normalized; thus it will be a family of vectors that will satisfy this criteria. Using any of these vectors will do. Additionally, some words in WordNet may not have an embedding. For these words, you can ignore them.

2.2 Run the code that finds the closest synsets for a selection of words that are not found in the WordNet vocabulary. Then, choose the of the selected words and answer the following questions for each one: Was the most similar

synset what you expected, or did it surprise you? Why do you think that synset was the most similar, based on what you know about WordNet, word embeddings, and the data that the embeddings were trained on?

3 Deliverables

Submit to GradeScope:

- Completed `SHW5.ipynb` notebook including:
  - `get_hyponym_terms`
  - `get_synset_locations`
  - `create_clusters`
  - `Answers to the questions on three out of vocabulary words`
- PDF of `SHW5.ipynb` notebook with each cell executed.