CSC485 Assignment 2

Nachiket Bhatt

November 2020

1 Question 0

1.1 (a)

[Code is submitted]

From the code I ran I found that the following result:

```
Synset with max depth is: Synset('rock_hind.n.01')

Routes from max depth synset to a root hyperonym:

Synset('rock_hind.n.01') at depth 15
Synset('hind.n.01') at depth 14
Synset('grouper.n.02') at depth 13
Synset('sea_bass.n.02') at depth 12
Synset('food_fish.n.01') at depth 11
Synset('fish.n.01') at depth 10
Synset('aquatic_vertebrate.n.01') at depth 9
Synset('vertebrate.n.01') at depth 8
Synset('chordate.n.01') at depth 7
Synset('animal.n.01') at depth 6
Synset('organism.n.01') at depth 5
Synset('living_thing.n.01') at depth 4
Synset('whole.n.02') at depth 3
Synset('object.n.01') at depth 2
Synset('physical_entity.n.01') at depth 1
Synset('entity.n.01') at depth 0

Synset('rock_hind.n.01') at depth 19
Synset('hind.n.01') at depth 18
Synset('grouper.n.02') at depth 17
Synset('sea_bass.n.02') at depth 16
Synset('serranid_fish.n.01') at depth 15
Synset('percoid_fish.n.01') at depth 14
Synset('spiny-finned_fish.n.01') at depth 13
Synset('teleost_fish.n.01') at depth 12
Synset('bony_fish.n.01') at depth 11
Synset('fish.n.01') at depth 10
Synset('aquatic_vertebrate.n.01') at depth 9
Synset('vertebrate.n.01') at depth 8
Synset('chordate.n.01') at depth 7
Synset('animal.n.01') at depth 6
Synset('organism.n.01') at depth 5
Synset('living_thing.n.01') at depth 4
Synset('whole.n.02') at depth 3
Synset('object.n.01') at depth 2
Synset('physical_entity.n.01') at depth 1
Synset('entity.n.01') at depth 0
```

2 Question 1

2.1 (d)

With more tokens to add, the signature set would have many more elements in it. These added tokens expand our horizon of hyperonyms having more than just a definition and some examples. By examining all the definitions and examples of the hyponyms, meronyms and holonyms, we get a better sense of how the hyperonym might be used. This gives us a better comparison than our previous version, making it more likey to choose a better option. This extension resulted in a accuracy increase of about 6%.

2.2 (f)

Let's call the sets the two vectors represent A and B.
The dot product of the two vectors would be just we the number of components which are 1 in both vectors. This is just the number of common elements in A and B i.e. their overlap.
This dot product is then divided by the product of the norms of the two vectors. The norm of vector of A would be the square root of the number of 1s in the vector i.e. the square root of the cardinality of A. Same goes for B.
Thus, we can say the CosSim of the two vectors would be:

$$\frac{Overlap(A,B)}{\sqrt{|A||B|}}$$

2.3 (h)

Using lower-cased versions caused a reduction in accuracy in all methods and different versions of lesk's algorithm.
This might be because lower-casing reduces the size of overlap with a signature which has lower and higher case words similar with the context.

This can be thought of as losing integral linguistic information since capitalization of words tell us about the type of word (proper noun or not), position in sentence etc. This vital information is lost when we lower-case everything.

3 Question 2

3.1 (a)

Sentences with the multiple of the same word but different sense are almost impossible for such algorithms to disambiguate. For example, the sentence "I will address the issue of the address". If we want to disambiguate the second "address", we see that it would point to the verb form of "address" but we know that this "address" refers to a noun. Even within the noun forms, they would not be able to distinguish between "address" as a location and "address" as a formal communications.
The main reason why they are so difficult to disambiguate using these methods can't disambiguate is that they see both instances of "address" as the same and don't look at it's dependencies with other parts of the sentence. Another thing is that they don't look at where the word is used in the sentence. This means that they can't actively differentiate between "address" coming after "will" and "address" coming after "the".

3.2 (d)

As pointed previously, if we want to disambiguate arbitrary sentences, we would still need to know which words to disambiguate. This is the biggest issue.
Another issue would be that even if we are given a word in a sentence to disambiguate, we might not have gathered enough vectors for the senses of that particular word to make a good prediction. In other words, if our training data is small, our predictions would be off.