CSCI 7000 - Assignment 1

Karan Praharaj

October 7, 2021

Problem 1

Token count for the given sentence (after lower-casing and removal of punctuation):

```
{'the': 7,
 'lion': 2,
 'king': 2,
 'takes': 1,
 'place': 1,
 'in': 1,
 'pride': 4,
 'lands': 2,
 'of': 4,
 'africa': 1,
 'where': 1,
 'a': 1,
 'rules': 1,
 'over': 1,
 'other': 1,
 'animals': 2,
 'as': 2,
 'dawn': 1,
 'breaks': 1,
 'all': 1,
 'are': 1,
 'summoned': 1,
 'to': 1,
 'rock': 1,
 'home': 1,
 'lions': 1}
```

Token count after words with frequency less than 2 are replaced with '<UNK>' :

```
{'<UNK>': 18,
 'the': 7,
 'lion': 2,
 'king': 2,
 'pride': 4,
 'lands': 2,
 'of': 4,
 'animals': 2,
 'as': 2}
```

The word co-occurence matrix is as follows:

	<UNK>	animals	as	king	lands	lion	of	pride	the
<UNK>	36.0	2.0	3.0	3.0	3.0	5.0	5.0	6.0	10.0
animals	2.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0	3.0
as	3.0	1.0	2.0	2.0	0.0	0.0	0.0	0.0	0.0
king	3.0	1.0	2.0	0.0	0.0	1.0	0.0	0.0	1.0
lands	3.0	0.0	0.0	0.0	0.0	0.0	1.0	2.0	2.0
lion	5.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0
of	5.0	1.0	0.0	0.0	1.0	0.0	0.0	4.0	5.0
pride	6.0	0.0	0.0	0.0	2.0	0.0	4.0	0.0	4.0
the	10.0	3.0	0.0	1.0	2.0	1.0	5.0	4.0	0.0

Even though technically, each word will be in its own context window, we did not keep count of the co-occurence of a word with itself. This can be changed to include these counts if we so desire.

Embeddings :

“the” : [10.0, 3.0, 0.0, 1.0, 2.0, 1.0, 5.0, 4.0, 0.0]

“lion” : [5.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0]

“king” : [3.0, 1.0, 2.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0]

Problem 2

The word the first embedding in the file represents : **“the”**

The number of words in the vocabulary : **40000**

The word the last embedding represents : **"sandberger"**

(Refer to code snippet below for working)

```
glove_embeddings = {}
with open("/Users/karanpraharaj/Downloads/glove/glove.6B.50d.txt", 'r') as f:
    for line in f:
        values = line.split()
        word = values[0]
        vector = np.asarray(values[1:], "float32")
        glove_embeddings[word] = vector

first_word = next(iter(glove_embeddings))
last_word = list(glove_embeddings)[-1]
```

```
first_word
```

```
'the'
```

```
last_word
```

```
'sandberger'
```

```
f"Total number of words in the vocabulary is : {len(glove_embeddings)}"
```

```
'Total number of words in the vocabulary is : 400000'
```

Problem 3

In this problem, we use sklearn's cosine_similarity function. Because, we have been asked to report cosine **distance** and not cosine similarity, we will subtract the similarity value obtained from 1. (Assuming $distance = 1 - similarity$)

1. Euclidean distance between *"lion"* and *"the"* based on word-word co-occurrence embeddings = **9.0**

   Cosine distance between *"lion"* and *"the"* based on word-word co-occurrence embeddings = **0.214**

```
lion_1 = np.array(df.loc['lion'])  # Co-occurrence based embedding for "lion"
the_1 = np.array(df.loc['the'])    # Co-occurrence based embedding for "the"

euc_dist = np.linalg.norm(lion_1 - the_1)
print(f"Euclidean distance between embeddings of 'the' and 'lion' from Problem 1 = {euc_dist}")

cosine_dist = cosine_similarity(lion_1.reshape(1,-1), the_1.reshape(1,-1))
print(f"Cosine distance between embeddings of 'the' and 'lion' from Problem 1 = {1-cosine_dist[0]}")
```

```
Euclidean distance between embeddings of 'the' and 'lion' from Problem 1 = 9.0
Cosine distance between embeddings of 'the' and 'lion' from Problem 1 = [0.21417472]
```

2. Euclidean distance between *"the"* and *"king"* based on word-word co-occurrence embeddings = **10.198**

   Cosine distance between *"the"* and *"king"* based on word-word co-occurrence embeddings = **0.319**

```
the_1 = np.array(df.loc['the'])  # Co-occurrence based embedding for "the"
king_1 = np.array(df.loc['king'])    # Co-occurrence based embedding for "king"

dist = np.linalg.norm(the_1 - king_1)
print(f"Euclidean distance between embeddings of 'the' and 'king' from Problem 1 = {dist}")

cosine_dist = cosine_similarity(the_1.reshape(1,-1), king_1.reshape(1,-1))
print(f"Cosine distance between embeddings of 'the' and 'king' from Problem 1 = {1-cosine_dist[0]}")
```

```
Euclidean distance between embeddings of 'the' and 'king' from Problem 1 = 10.198039027185569
Cosine distance between embeddings of 'the' and 'king' from Problem 1 = [0.31945535]
```

3. Euclidean distance between *"lion"* and *"the"* based on GloVe embeddings = **4.989**

   Cosine distance between *"lion"* and *"the"* based on GloVe embeddings = **0.544**

```
lion_2 = glove_embeddings['lion']  # GloVe embedding for "lion"
the_2 = glove_embeddings['the']    # GloVe embedding for "the"

euc_dist = np.linalg.norm(lion_2 - the_2)
print(f"Euclidean distance between embeddings of 'the' and 'lion' from Problem 2 = {euc_dist}")

cosine_dist = cosine_similarity(lion_2.reshape(1,-1), the_2.reshape(1,-1))
print(f"Cosine distance between embeddings of 'the' and 'lion' from Problem 2 = {1-cosine_dist[0]}")
```

```
Euclidean distance between embeddings of 'the' and 'lion' from Problem 2 = 4.988907337188721
Cosine distance between embeddings of 'the' and 'lion' from Problem 2 = [0.5444176]
```

4. Euclidean distance between *"the"* and *"king"* based on GloVe embeddings = **4.797**

   Cosine distance between *"the"* and *"king"* based on GloVe embeddings = **0.428**

```
the_2 = glove_embeddings['the']    # GloVe embedding for "the"
king_2 = glove_embeddings['king']    # GloVe embedding for "king"

euc_dist = np.linalg.norm(the_2 - king_2)
print(f"Euclidean distance between embeddings of 'the' and 'king' from Problem 2 = {euc_dist}")

cosine_dist = cosine_similarity(the_2.reshape(1,-1), king_2.reshape(1,-1))
print(f"Cosine distance between embeddings of 'the' and 'king' from Problem 1 = {1-cosine_dist[0]}")
```

```
Euclidean distance between embeddings of 'the' and 'king' from Problem 2 = 4.796682834625244
Cosine distance between embeddings of 'the' and 'king' from Problem 1 = [0.42777586]
```