Assignment 1.2

Practical Deep Learning for Language Processing

Dr. Aseem Behl
School of Business and Economics
University of Tübingen
Friday, 25. November 2022

Word Embeddings with Word Co-occurrence Matrix

In this exercise, you will practice building a word co-occurrence matrix which can be used to represent words as vectors as discussed in Chapter_04 of the lecture. You will need to implement the following steps in this assignment:

Feel free to use the following third party python packages: `sklearn, nltk, numpy, pandas`

Step 1: Load the *Brown Corpus of Standard American English* using `brown.txt` provided to you on ILIAS.

Step 2: Write a helper function, `get_word_frequencies()` which will take as input the text corpus and return the frequencies of occurrence of each linguistic token (or word) in our brown corpus.

Step 3: Write a second helper function `create_vocab()` which will take as input the word frequencies returned by `get_word_frequencies()`. In this function, you will select the `|V|` tokens with the highest frequency and return a vocabulary data structure which will store the mapping from the `|V|` linguistic tokens to their corresponding index. Here `|V|` is a hyper-parameter which defines the size of your vocabulary. For this assignment, you can pick `|V|` to be `20000` words.

Step 4: To compute word co-occurrences, we will consider neighbourhood of size 1, i.e., one word to the left and one word the the right. Implement a third helper function `windowizer()` which will take as input the text corpus and return a list of all two word windows. For example if the input text was: `'The Fulton County Grand Jury said Friday…'`. The output would be

the list: `['The Fulton', 'Fulton County', 'County Grand', 'Grand Jury', 'Jury said', 'said Friday', 'Friday an',…]`

Step 5: Next using the above three helper functions construct the word co-occurrence embedding matrix for the brown corpus. *Hint*: `sklearn.feature_extraction.text.CountVectorizer` could be helpful to compute the co-occurrence counts.

Step 6: Finally, implement a function, `most_similar_words()` which will take as input the embedding matrix, the vocabulary and a test word. Your function should return the top 5 words in your vocabulary which are most similar to the test word as measured by cosine similarity.

The task is worth **6 points** and is due on Saturday**, December 03, at 10:00 PM CET**. For submission, you only need to upload your notebook to ILIAS. Run all cells, and do not clear out the outputs, before submitting. In order to receive credit for the assignment, please be prepared to present your solutions during the lecture on **Tuesday, December 06**. You may be asked other questions from the lecture material related to the assignment.