CSE 556, Natural Language Processing
Monsoon, 2019
Assignment 5
Deadline: 23:59 hrs, 6th November, 2019

Instructions:

- **Please don't copy from internet or any other student. Plagiarism will thoroughly be checked on all the submitted files. Refer: Academic Dishonesty Policy**
- **Allowed Programming language: Python**
- **Use classroom discussion for any doubt.**
- **Total points for this assignment is 50.**
- **Dataset can be obtained from here.**

Task 1: Cosine Similarity [25 points]

You will be creating a question-answering system and determine its accuracy.
Question-answering system requires a model trained on some dataset and a set of questions serving as the query to model.

Part-1: Model

Create a term-document matrix using the **data.txt** file. The dataset contains around 1000 sentences from science text book each on a separate line. Treat each sentence as a single document. The term-document matrix should be created using **tf-idf** for the term.

tf–idf or **term frequency–inverse document frequency**, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling.
For this assignment td-idf should be calculated as:

$$(1 + \log(1 + f_{t,d})) \cdot \log\left(\frac{N}{1 + n_t}\right)$$

[source]

Part-2: Query

Use **test.jsonl** file to obtain the questions. Each line contains 1 question in json format along with 4 options. To form a query, combine each question with it's option, ie, (Q_i + O_i).

Part-3: Evaluation

For each query, find the **cosine similarity** between the query and each of the documents and use the similarity score of the most matching document as the score of your system for the query. Repeat this for all the options and your answer for a particular question should be the (Q_i + O_i) combination with the maximum score.

Scoring function:
Note: Since the similarity score can be same for more than 1 query ((Q_i + O_i) pairs), your score for a question should be calculated using below pseudo-code:

If answerKey(available in the test file) ϵ set of option with maximum similarity score:
    score(Q_i) = (1) / (# option with maximum similarity score)
else:
    score(Q_i) = 0

Repeat the above for all questions available in the test file.

Accuracy:
Report you accuracy, which should be calculated as:
Accuracy = (sum of all score) / (# question)

Task 2: Doc2Vec [25 points]

Use the same dataset of 1000 sentence to train your Doc2Vec model and repeat the above steps as in task 1, ie,

- Query formulation, (Q_i + O_i).
- Find the most likely document that contains the answer for the question using document similarity score using Doc2Vec model.
- Use the same scoring function and find the accuracy.
- **Note:**
    - You may need to normalize the score.
    - No need to implement the algorithm for Doc2Vec. You are allowed to use **gensim** library for this.

What to Submit?

A zip/tar file containing:

- **Running code.**
- **Trained model**
- **Report. Should have:**
    - **Preprocessing steps used.**
    - **State all you assumptions.**
    - **Observations, including:**
        1. **Your understanding of Doc2Vec model and how it differs from Word2Vec.**
        2. **Kind of documents returned using the 2 models.**
        3. **Similarity score between query and document.**
        4. **Inferences the type of option receiving high similarity score.**
    - **Accuracy values**