CSE508 Information Retrieval
Winter 2024
Assignment-1

Due Date: Feb 7, 2024 ; 23:59 **Max. Marks:** 100

Instructions:

1. The assignment is to be attempted individually.
2. The proposed solutions will be evaluated during your code demo and viva.
3. Institute plagiarism policy will be strictly followed.
4. The programming language allowed is Python.
5. Ensure that your code is thoroughly documented for clarity and understanding.
6. You can utilize libraries such as NLTK and BeautifulSoup for data preprocessing in your Python code.
7. You are required to use version control via GitHub:
    a. Make a GitHub repository with the name:
    CSE508_Winter2024_A1_<Roll_No.>.
    b. Add your assignment TA as a contributor. The TA assigned (along with their GitHub handle) to you for this assignment will be released on classroom.
8. You must make a detailed report with the name **CSE508_Winter2024_A1_<Roll_No>_Report.pdf** with a brief overview of your approach, methodologies, assumptions, and results for each problem.
9. Steps for submission:
    a. A zipped folder **CSE508_Winter2024_A1_<Roll_No.>** consisting of all your code files, dumped files and **Report.pdf**
    b. A text file **CSE508_Winter2024_A1_<Roll_No.>.txt** consisting of the link to your GitHub repository.
10. Use the dataset provided here for problems in the assignment. - Link to Dataset - Dataset [999 files].
11. If it has been mentioned to code a solution from scratch, using any library is strictly not allowed.

Sample Text File from the Dataset : *file1.txt*
Loving these vintage springs on my vintage strat. They have a good tension and great stability. If you are floating your bridge and want the most out of your springs than these are the way to go.

Q1. Data Preprocessing [20 Marks]

1. Perform the following preprocessing steps on each of the text files in the dataset linked above.
   a. Lowercase the text
   b. Perform tokenization
   c. Remove stopwords
   d. Remove punctuations
   e. Remove blank space tokens
2. Print contents of 5 sample files before and after performing each operation. Remember to save each file after preprocessing to use the preprocessed file for the following tasks.

Q2. Unigram Inverted Index and Boolean Queries [40 Marks]

1. Create a unigram inverted index (from scratch; No library allowed) of the dataset obtained from Q1 (after preprocessing).
2. Use Python's **pickle** module to save and load the unigram inverted index.
3. Provide support for the following operations:
   a. T1 **AND** T2
   b. T1 **OR** T2
   c. T1 **AND NOT** T2
   d. T1 **OR NOT** T2
4. Queries should be generalized i.e., you should provide support for queries like T1 **AND** T2 **OR** T3 **AND** T4 ….
5. **Input format:**
   a. The first line contains **N** denoting the number of queries to execute
   b. The next **2N** lines contain queries in the following format:
      i. Input sequence
      ii. Operations separated by comma
6. **Output Format:**
   a. **3N** lines consisting of the results in the following format:
      i. Query X
      ii. Number of documents retrieved for query X
      iii. Name of the documents retrieved for query X (as *file1.txt*)
7. Perform preprocessing steps (from Q1) on the input sequence as well.
8. Sample Test Case: [Please note that the output values are dummy values; The test case is given just to comprehend the format.]

   a. **Input:**
      2
      Car bag in a canister
      OR, AND NOT
      Coffee brewing techniques in cookbook
      AND, OR NOT, OR

   b. **Output:**
      Query 1**:** car **OR** bag **AND NOT** canister
      Number of documents retrieved for query 1: **3**
      Names of the documents retrieved for query 1: **a.txt, b.txt, c.txt**
      Query 2: coffee **AND** brewing **OR NOT** techniques **OR** cookbook
      Number of documents retrieved for query 2: **2**
      Names of the documents retrieved for query 2: **d.txt, e.txt**

9. You must run your code during the demo and strictly follow the Input/Output format.

Q3. Positional Index and Phrase Queries [40 Marks]

1. Create a positional index (from scratch; No library allowed) of the dataset obtained from Q1.
2. Use Python's **pickle** module to save and load the positional index.
3. **Input Format:**
   a. The first line contains **N** denoting the number of queries to execute
   b. The next **N** lines contain phrase queries
4. **Output Format:**
   a. **2N** lines consisting of the results in the following format:
      i. Number of documents retrieved for query X using positional index
      ii. Names of documents retrieved for query X using positional index
5. Perform preprocessing steps (from Q1) on the input sequence as well. Assume the length of the input sequence to be <=5.
6. **Sample Test Case:** [Please note that the output values are dummy values; The test case is given just to comprehend the format.]
   a. **Input**
      2
      Car bag in a canister
      Coffee brewing techniques in cookbook

b. **Output**

   Number of documents retrieved for query 1 using positional index:
   Names of documents retrieved for query 1 using positional index: **a.txt, b.txt**
   Number of documents retrieved for query 2 using positional index: **2**
   Names of documents retrieved for query 2 using positional index: **a.txt, b.txt**

7. You must run your code during the demo and strictly follow the Input/Output format.