CSE508 Information Retrieval
Winter 2023
Assignment-1

Due Date: Feb 16, 2023, 23:59 **Max. Marks:** 100

Instructions:

1. The assignment is to be attempted in groups of max 3 members.
2. Each group member must do at least one task. All members should know the working of all the tasks. This will be evaluated during your code demo and viva.
3. Institute plagiarism policy will be strictly followed.
4. Programming language allowed: Python.
5. Your code should be well documented.
6. You are free to use libraries like NLTK, BeautifulSoup for data preprocessing.
7. You are required to use version control via GitHub:
    a. Make a GitHub repository with the name: **CSE508_Winter2023_A1_<Group_No.>**.
    b. Add your assignment TA as a contributor. The TA assigned (along with their GitHub handle) to your assignment group for this assignment can be found here. c. Contribution of each member will be monitored via git commits.
8. You must make a detailed report with the name **Report.pdf** covering your methodologies, assumptions, and results.
9. Submission:
    a. A zipped folder **CSE508_Winter2023_A1_<Group_No.>** consisting of all your code files, dumped files and **Report.pdf**
    b. A text file **CSE508_Winter2023_A1_<Group_No.>.txt** consisting of the link to your GitHub repository.
10. Only one member from a group needs to submit.

Dataset Link: Dataset [1400 files]

Q1. Data Preprocessing

(i) Relevant Text Extraction [2 Marks] For each file, extract the contents between the <TITLE>...</TITLE> and <TEXT>...</TEXT> tags and concatenate the 2 strings using blank space. Discard the rest of the text and save the string obtained above in the same file. [Do NOT change filename].

Example:-
Final contents of the file **carnfield003**: *the boundary layer in simple shear flow past a flat plate . the boundary-layer equations are presented for steady incompressible flow with no pressure gradient .*

Perform this on all 1400 files. Print contents of 5 sample files before and after performing the operation.

(ii) Preprocessing
Carry out the following preprocessing steps on the dataset obtained above: **[8 Marks]** 1.
Lowercase the text
2. Perform tokenization
3. Remove stopwords
4. Remove punctuations
5. Remove blank space tokens

Print contents of 5 sample files before and after performing EACH operation.

Q2. Boolean Queries [20 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 **AND NOT** T2
    c. T1 **OR** 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 **OR** T4 ….
5. You are also required to compute the minimum number of comparisons done to execute the query [only where merging is required]
6. **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

7. **Output Format:**
    a. 4N lines consisting of the results in the following format:
        i. Query X
        ii. Number of documents retrieved for query X
        iii. Names of the documents retrieved for query X
        iv. Number of comparisons required for query X

8. Perform preprocessing steps (from Q1) on the input sequence as well. 9. 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**
Number of comparisons required for query 1: **23**
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**
Number of comparisons required for query 2: **13**

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

Q3. Phrase Queries

(i) Bigram inverted index [30 Marks] 1. Create a bigram inverted index (from scratch; No library allowed) of the dataset obtained from Q1.
2. Use Python's **pickle** module to save and load the bigram inverted index.

(ii) Positional index [35 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.

(iii) Compare and comment on your results using (i) and (ii). [5 Marks]

1. **Input Format:**
    a. The first line contains **N** denoting the number of queries to execute
    b. The next **N** lines contain phrase queries
2. **Output Format:**
    a. **4N** lines consisting of the results in the following format:
        i. Number of documents retrieved for query X using bigram inverted index ii. Names of documents retrieved for query X using bigram inverted index iii. Number of documents retrieved for query X using positional index iv. Names of documents retrieved for query X using positional index
3. Perform preprocessing steps (from Q1) on the input sequence as well. Assume the length of the input sequence to be <=5.
4. **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 bigram inverted index: **3**
        Names of documents retrieved for query 1 using bigram inverted index: **a.txt, b.txt, c.txt**
        Number of documents retrieved for query 1 using positional index: **2**
        Names of documents retrieved for query 1 using positional index: **a.txt, b.txt**
        Number of documents retrieved for query 2 using bigram inverted index: **4**
        Names of documents retrieved for query 2 using bigram inverted index: **a.txt, b.txt, c.txt, d.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**
5. You must run your code during the demo and strictly follow the Input/Output format.