Group 13

Work contributions:
We have discussed the project among ourselves and we have developed the workflow for the same but for developing the codes, we have divided the work among the group as follows:
Task 1A (Building Index) - Sanskar Patni
Task 1B (Query Preprocessing), Readme - Abhishek S Purohit
Task 1C (Boolean Retrieval) - Yash Jain

Short description of approach:

Task 1A (Building Index)
Created a function to preprocess sentence which removes stop words, punctuation marks and performs lemmatization to generate tokens from the corpus.For the preprocessing we used libraries like regex for punctuation removal and nltk library for stop words removal and lemmatization.
Then used csv.dictreader function of python to store the csv mapping of cord_id and paper_id to a python dictionary and read all the documents using glob library.
When we read each document, we first preprocessed the abstract and then used the tokenized words to generate inverted index. For creating the inverted index, we decided to store inverted index as a dictionary where each word will be mapped with it's corresponding postings list. So for every word in that preprocessed sentence we checked if the corresponding postings list had the cord_id of document, if not we inserted it in the posting list.
Using this logic the posting list id were not guaranteed to be in sorted order as is required for boolean retrieval, so we sorted each of the posting list. As this whole process took a very long time to complete it was not possible to run this code again and again, thus we saved the inverted index to a bin file using pickle.

Task 1B (Query Preprocessing)
Script will first open a file "queries_13.txt" in write mode, which will create a new file (pointed by outputFile in the code) (or overwrite the file, if it already exists) in the main code directory for storing the processed queries. Further, we will read the raw queries file located at the path provided by the user as argument while invoking the script. We have developed a function with the help of nltk and regex libraries available in python to preprocess input text which involves removing stop words and punctuation marks and further performing lemmatization (without POS tags) to generate tokens for the given text. For reading the raw queries file, we have used csv.reader() function which takes file pointer as input. Later we iterated upon all the query lines (except the first line which contains the table headings) and at each iteration we are processing the raw query text by passing it to the preprocess function, following which we are writing the processed query along with its query-id to the output file as instructed in the problem statement. Once we reach EOF, our loop gets terminated and then our script safely closes both the files before returning back the control.
So, in this way our code generates the processed query file in the main directory which will be used in subsequent parts.

Task 1C (Boolean Retrieval)

Firstly, using the pickle library, the inverted indexes are loaded from the model using the “pickle.load” function in the library in order to convert the binary file of the inverted index into postings list.

Now, loading the “queries_13.txt”(path will be given in the command line) file which contains all the queries for which we want to find the results.Now, for each query in the text file we generate the tokens from the query for which the postings list is present in the inverted index and according the to the question,we have to find the intersection of the documents present in postings list of all tokens present in the query.

In order to implement the logic for intersection, for each query we first sort all the tokens in the query according to the size of the length of their postings list which is done using the “sorted” functions of the functools library.

Now,the script traverses through the tokens which are sorted according to the size of their postings list and merge them to give the result for each query.The “merge” function takes two sorted postings list and gives the intersection of the postings list as result using the two pointer approach.

For each query in the text file, the resultant list which is generated after the traversal of all the tokens in the query and the result for each query is stored in the “Assignment1_13_results.txt” file.