Birla Institute of Technology and Science, Pilani

Information Retrieval (CS F469)

Second Semester 2016-17

Assignment - 1

Due Date : 23-02-2017 **Total Marks: 30**

In this assignment, you are required to implement a **text-based search engine** (based on either **Boolean Retrieval** or **Vector Space Model**). The assignment has several stages where you will build the indexing component, which will take a large collection of text and produce a searchable, persistent data structure. Thereafter, you will add the searching component, according to Boolean Retrieval or Vector Space Model.

The assignment is a group assignment and has to be done in **groups** comprising of **4-5 members**. Please note that a group should not contain less than 4 members. All the team members are expected to contribute to all aspects of the assignment: design, implementation, documentation, and testing. It is recommended that each group maintains a **github repository** for this assignment. You should commit in small logical commits with proper commit messages. You can use these links to Git Tutorial and Git Documentation.

Languages which can be used - C, C++, Java, Python.

Corpus - Any Corpus is allowed. Here are some links for some of the popular corpus collections-

- Stanford Large Network Dataset Collection
- Network Data (Social Computing, AZU)
- NLTK Corpora
- Reuters Corpus

Below is the outline of the assignment in stages-

1) **Tokenization**

   All documents in the corpus are supposed to be tokenized. You can use standard tokenization libraries which are available in various languages such as - NLTK(Python), Stanford Tokenizer(Java),Boost(C++), etc.

2) **Normalization**

   All tokens processed from the previous stage are supposed to be normalized. You can choose either Lemmatization or Stemming. Libraries can be used to achieve the above. Most of the languages have packages which have Stemming and Lemmatization Algorithms implemented.

3) **Building Index**

   An inverted index has to be implemented for the processed tokens. Use appropriate data structure for building the index however, you are restricted from using any kind of external libraries in this stage.

4) **Spelling Correction**

   You can add a spelling correction phase which will correct any erroneous spellings in user queries.

5) **Retrieval of Documents**

   In this stage you are supposed to implement the search module which would process the query entered by the user and retrieve the results from the corpora using the produced data structures. As mentioned

before you can use either Boolean Retrieval Model or Vector Space Model to rank the documents by choosing appropriate weighting function. Usage of any library for this stage is not allowed.
The choice of model should be justified in the design document.
NOTE - For each query return at least 10 and at most 20 results.

6) Performance Metrics

The following metrics must be displayed along with search results -

- **F-Measure** (To be displayed with each query)**:** Defined as the harmonic mean of Precision and Recall. You can refer these links: Wikipedia, Introduction to Information Retrieval
- **Mean Reciprocal Rank (MRR):** Defined as the average of inverse of the rank of the first correct answer for a sample of queries. You can refer these links: Wikipedia
- **Mean Average Precision (MAP):** Defined for a set of queries as the mean of the average precision scores for each query. You can refer these links: Introduction to Information Retrieval

Note- For MRR and MAP, calculate the parameters using a minimum of 20 queries. Also include the mean F-measure for those queries. Include the queries you used in the report.
These are very essential and would be a major part of the group evaluation.

Deliverables

1. **Code** - Should be well documented. The purpose and intent of each method, class and modules should be mentioned.
2. **Design Document** - Should justify each and every aspect of the implementation including the data structures and algorithms used. You can also elaborate the workflow of your search engine. List down at least 5 queries and their corresponding results along with the performance metrics.
3. **README file** - List down all the steps for compiling your code and running the search engine for any alternate corpus that we would like to test on it. Also list down all the assumptions that you have made while implementing this search engine (if any), along with the Github Repository link.

Interface

The search engine can have a simple command line interface. But the driver code, when run, should ask for entering a number corresponding to the action for which the user wants to display the results, as follows-

- Entering "1" should display the result of tokenization and normalization of the query.
- Entering "2" should display the result of the spell correction of the query.
- Entering "3" should fetch the documents for the query.

Any kind of extra efforts made by the group to improve the performance/experience would be also considered suitably. For example, adding support for wildcard queries, proximity queries or considering a dynamic corpus, etc.

Submission Details

Make a Zip of all the above mentioned deliverables and mail it to lavika.goel@pilani.bits-pilani.ac.in before the deadline along with the subject line "IR Assignment 1: Group no. __(mention your group no.)". Also mention all the group members' names and IDs in the design document as well as in the email.

You are also expected to demo your program including an explanation of your evaluation results to us as per the schedule which will be made available.

Evaluation Scheme

Task	Marks
Tokenization and normalization	5
Building index	7
Spell Correction	3
Documentation with performance metrics	5
Viva	5
Innovation / Novelty	5