BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

SECOND SEMESTER 2019-2020 INFORMATION RETRIEVAL CS F469

Assignment 2

Due Date: 29/02/2020 **Weightage: 10%**

The objective of this assignment is to build a vector space-based information retrieval system. Implementing a working system will help you in better understanding of how actual IR systems work and what are some practical issues faced during the building of an IR system.

Corpus details: You are required to use the same corpus as used for Assignment 1. You are free to choose any one or more files from the shared corpus. A single file in the corpus contains multiple documents. These documents have a start and the end tag as:

start tag: <doc id="Document id" url="Wikipedia url" title="Document title">
end tag: </doc>

All the text present in between the start and the end tag is associated with the same document.

Programming language: You are required to implement this assignment in **Python** programming language only.

For this assignment, you can use an in-memory based index construction algorithm. A simple way would be to use a python dictionary data structure, with keys as a term and a list as doc id. However, you are free to use any other data structure. The primary focus of the assignment is on the ranking of documents wrt queries.

The assignment is divided into two parts, as described below:

Part 1: A working ranked retrieval based IR system and its evaluation.

Here you need to build an IR system with the following characteristics:

1. The vector space model should be used for computing the score between document and query.
2. Use the lnc.ltc scoring scheme (based on SMART notation).
3. The queries should be free-text queries.
4. Do not remove stop words. Do not perform stemming/lemmatization and normalization. Punctuations can be removed.

After the system is built, you need to evaluate the system on at least 10 multi-term queries. The queries should be selected such that it covers several cases, for example, common nouns, proper nouns, rare terms, ambiguous terms, etc. For each query, you have to evaluate the top 10 documents retrieved, and mark them manually, whether they are relevant to the query or not as per the following format:

Query	Top K Documents	Score	Is the document relevant to the query?
sample query	Document title 1		
	Document title 2		
	Document title 3		
	Document title 4		
	Document title 5		
	Document title 6		
	Document title 7		
	Document title 8		
	Document title 9		
	Document title 10		

Part 2: Improve the retrieval and ranking for the documents. You have to propose and implement two improvements.

For each improvement, answer the following questions briefly:

1. What is the issue with the IR system built in part 1?
2. What improvement are you proposing?
3. How will the proposed improvement address that issue?
4. A corner case (if any) where this improvement might not work or can have an adverse effect.
5. Demonstrate the actual impact of the improvement. Give three queries, where the improvement yields better results compared to the part 1 implementation.

Note: There is no need to build a browser/gui interface. Display of the output on the terminal/notebook would suffice.

Implementation guidelines:

There should be two components in the code:

1. The index creation code: One or more code files that will be used to create the inverted index as well as all necessary structured information that you want to extract from the corpus. This code should store the inverted index (and/or any other data that you want to extract from the corpus) on one or more files on the disk (in any format).

2. A single file names test_queries.py, the will take as an input a query and paths for the stored inverted index (and/or any other data) and should output the top K documents. This file should never read the text corpus.

Deliverables:

1. Well commented code. The purpose and intent of each method, class and module should be mentioned appropriately.
2. Report: A report describing all important assumptions made for implementing, limitations, algorithms used etc. It must include the answers to part 2 questions and part 1 evaluation results.
3. Readme file having all the steps for running your code.

For you reference, a sample solution for part 2 can be:

1. The IR system built in part 1 does not work well when a query term is misspelled.
2. We are proposing to use a spelling corrector on the query.
3. The proposed improvement will help because: …
4. We demonstrate the impact using these queries as reference:
    a. A misspelled query: Documents retrieved earlier vs documents retrieved after improvement.

FAQ:

Q1: How will this assignment be evaluated?
A1: Three parameters:

1. The working code.
2. The report.
3. Your understanding about the code, report and about the overall working of the system.

Q2: Can I use a X library?
A2: You are free to use any library which is open source.

Q3: What is the weightage of part 1 and part 2?
A3: 50% weightage for part 1, 50% weightage for part 2.

Submission guidelines: Will be posted later.