Recuperação de Informação / Information Retrieval

2017/2018 MIECT/MEI, DETI, UA

Assignment 3

Submission deadline: **21 November 2017**

For this assignment, you will create a weighted (tf-idf) indexer and ranked retrieval system. Use the same corpus as in assignment 1.

1. Create an indexer class that applies tf-idf weights to terms.
   Reuse the tokenizer that incorporates stemming and stopword filtering from Assignment 1.
   Save the resulting index to a file using the following format (one term per line):
   term,doc_id:term_weight,doc_id:term_weight,…

2. Create a class that implements a ranked retrieval method.

3. Process the queries (file 'cranfield.queries.txt') and retrieve the results for each query.
   Write the results, sorted by document score, to a text file using the same format as in Assignment 2.

4. Using the relevance scores (*gold standard*) provided, calculate the following evaluation and efficiency metrics for this implementation and for the implementations in the second assignment:

   a. Precision
   b. Recall
   c. F-measure
   d. Mean Average Precision
   e. Mean Precision at Rank 10
   f. Mean Reciprocal Rank

   i. Query throughput
   ii. Median query latency

Note:

Your assignment will be evaluated in terms of: modelling, class diagram, code structure, organization and readability, correct use of data structures, submitted results, and report. See suggestions and submission instructions below.

Suggestions:

- Write **modular** code
- Favour **efficient** data structures
- Add **comments** to your code
- Follow the **submission instructions**

Submission instructions:

- To manage your project please use **Maven** (preferably) or Netbeans
- At each submission, include a small **Report** including:
    - Your project's **class diagram**
    - A description of each class and main methods, identifying where these are called
    - A block diagram and a high-level (but sufficiently detailed) description of the overall processing pipeline (data flow diagram)
    - Complete instructions on how to run your code, including any parameters that need to be changed
    - A list of any external libraries that are needed to run the code
    - Efficiency measures: total indexing time; maximum amount of memory used during indexing; total index size on disk
    - A short commentary/assessment of your own work, describing features or implementation decisions that you consider the most relevant/positive (or otherwise)
- Make sure you **include your name and student number** in the code and in the report.
- Make sure all your programs compile and run correctly.
- Submit your assignment by the due date using Moodle.