Recuperação de Informação / Information Retrieval

2021/2022 MEI/MIECT, DETI, UA

Assignment 1

Submission deadline: **19 November 2021**

For this assignment you will create a document indexer using the SPIMI approach. Implement two alternative indexing limits in SPIMI: number of postings and amount of memory used.

The corpus for this assignment is the 'Amazon Costumer Reviews Dataset', described here https://s3.amazonaws.com/amazon-reviews-pds/readme.html

1. Create a document parser that iterates over the collection (corpus) of document and returns, in turn, the contents of each document.
   For this assignment consider only the `product_title`, `review_headline` and `review_body` as content (indexed) fields and use the `review_id` as the document identifier.

2. Create a tokenizer with the following processing of tokens:
   i. A minimum length filter that removes tokens with less than a default number of characters. Allow the user to disable the filter or set another value;
   ii. A stopword filter using a default list. Allow the user to disable the filter or use a different list;
   iii. Porter stemmer (from snowball or NLTK). Allow the user to disable the filter.

3. Index, separately, each of the files listed below and gather the following statistics:
   a) Total indexing time.
   b) Total index size on disk.
   c) Vocabulary size (number of terms).
   d) Number of temporary index segments written to disk (before merging).
   e) Amount of time taken to start up an index searcher, after the final index is written to disk. Note: consider the minimum data required by a searching mechanism. This light searcher should expect a query term and return the term document frequency (number of document where the term occurs).

Files to use:
amazon_reviews_us_Digital_Video_Games_v1_00.tsv.gz (26.2 MB)
amazon_reviews_us_Digital_Music_Purchase_v1_00.tsv.gz (241.8 MB)
amazon_reviews_us_Music_v1_00.tsv.gz (1.4 GB)
amazon_reviews_us_Books_v1_00.tsv.gz (2.6 GB)

Instructions:

- **Modelling**, code **structure**, **organization** and **readability** will be considered when grading your project
- **Comment** your code; and make sure you include your name and student number
- Write **modular** code
- Favour **efficient** data structures
- Use **parameters**, preferably through the command line
- Make sure all your programs run correctly
- Submit your assignment by the due date using Moodle