Recuperação de Informação / Information Retrieval

2017/2018 MIECT/MEI, DETI, UA

Assignment 1

For this assignment you will create a simple document indexer. This should consist of a corpus reader, document processor, tokenizer, and indexer.
The corpus for this assignment is available on the elearning platform.

1. Create a corpus reader class that returns, in turn, the contents of each document in a collection (corpus). The document should be processed to ignore any irrelevant sections and clean any existing tags (e.g. xml markup).
   Your implementation should be modular and easily extended/adapted to other corpora structures.

2. Create a tokenizer class that returns the tokens of an input text. In doing this, pay particular attention to characters that need special handling ('.'; ','; '-'; etc.).
   Integrate the Porter stemmer (http://snowball.tartarus.org/download.html) and a stopword filter. Use this list as default: http://snowball.tartarus.org/algorithms/english/stop.txt

3. Create an indexer and index the corpus using a suitable data structure defined by you. Apply stemming and stopword filtering and save the resulting index to a file using the following format (one term per line): term,doc id:term freq,…

4. Create a simple tokenizer that splits on whitespace, lowercases tokens, removes all non-alphabetic characters, and removes stopwords. Answer the following questions:
   a. What is your vocabulary size?
   b. What is your vocabulary size using the simple tokenizer?
   c. For each tokenizer, list the ten first terms (in alphabetic order) that appear in only one document.
   d. For each tokenizer, list the ten terms with higher document frequency.

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 data flow diagram and description of the overall processing steps
    - 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.