CE306/CE706 - Information Retrieval
Assignment 1: Indexing for Web Search

Alba García Seco de Herrera

5th February 2020

Plagiarism

You are reminded that this work is for credit towards the composite mark in CE306/CE706, and that the work you submit must therefore be your own. Any material you make use of, whether it be from textbooks, the Web or any other source must be acknowledged as a comment in the program, and the extent of the reference clearly indicated.

The Context of your Task

Here are some snippets extracted from an open job ad for a *Software Development Engineer* at Amazon.[1]

... Alexa Search is part of our ongoing Alexa Information efforts focused on reinventing information extraction and retrieval for a voice-forward, multi-modal future. We are looking for creative managers to help design, develop, and deploy systems that collect and process data at Web scale; [...] to improve access to and retrieval of these derivatives in the context of Alexa user experiences and other Amazon services.

Successful candidates will build and optimize scalable systems for processing and managing Web-scale repositories and will be involved in all stages of text processing and knowledge distillation, from collection, to processing and cleaning, to extraction and verification...

That looks exactly like the profile I would like you to have when you graduate!

The Task

Your task is to apply your IR skills to build a processing pipeline that turns a Web site into structured knowledge (thus enhancing your chances of getting the job outlined above). Your system should take HTML pages as input, process them using the kind of techniques that we have been looking at in the module, and output an index of terms identified in the documents.

This assignment comes in stages. Marks are given for each stage. You may choose not to attempt some stages. You might also implement a system that does not strictly follow the stages but will work in the same way. The stages are as follows:

- **Engineering a Complete System (10%)** The system you develop must be able to read Web pages from a specified set of URLs and produce appropriately formatted output. The Web pages should be processed one at a time using the steps outlined below. The final system should have control over all the individual components so that there is a single call and all the steps outlined below will be performed.
- **HTML Parsing (10%)** Before the text can be analysed it is necessary to get rid of the HTML tags. The result will be plain text. Note that if you simply delete all HTML tags, you will lose information such as meta tag keywords. Use an appropriate tool to perform this task.

[1] https://www.amazon.jobs/en/jobs/1027369/software-development-engineer

- **Pre-processing: Sentence Splitting, Tokenization and Normalization (10%)** The next step should be to transform the input text into a normal form of your choice. This should include the identification of sentences, bullet points and cells in tables.
- **Part-of-Speech Tagging (10%)** The input should be tagged with a suitable part-of-speech tagger, so that the result can then be processed in the next steps.
- **Selecting Keywords (20%)** One aim of your system is to identify the words *and* phrases in the text that are most useful for indexing purposes. Your system should remove words which are not useful, such as very frequent words or stopwords, and identify phrases suitable as index terms. Apply *tf.idf* as part of your selection and weighting step.
- **Stemming or Morphological Analysis (10%)** Writing word stems to the database rather than words allows to treat various inflected forms of a word in the same way, i.e. *bus* and *busses* refer to exactly the same thing even though they are different words.

You will have noticed that the percentages above only add up to 70%. This is because one of the important aspects of the project is that your work should be well documented and your code well commented. **30% of your mark will come from this.** In addition to the actual code you should submit:

- A description of your implementation: what the code does, and the software you used
- Output produced by a run of your system when applied to these two Web pages:

  `http://www.multimediaeval.org/mediaeval2019/memorability/`
  `https://sites.google.com/view/siirh2020/`

- Output produced by *each* stage of the processing pipeline for each of the two files, i.e. in the suggested staged architecture outlined above this would be output produced by the HTML parser, followed by the output of the sentence splitter, the tokenizer etc. Each stage should produce a separate file (however, to calculate weights such as *tf.idf* for the final index terms you will have to consult data derived from *all* documents and it is up to you how exactly you include that step in your processing pipeline).
- A *short* discussion of your solution focussing on functionality implemented and possible improvements and extensions.

You may work in pairs. If you do, you each need to submit the same report (please include information about which two reports should be treated as a pair). Both members of a pair will get the same mark unless there is reason to do otherwise.

You can implement your system either on the Linux or the Windows machines. Python, Java, Perl, C/C++, and shell scripts are good choices for this project, but you are by no means restricted to those languages. Identify suitable open-source tools that help you building your pipeline.

Submission

The assignment, which counts for **20%** of the overall mark, should be submitted as a single *zip file* via the electronic submission system by **Friday, 21 February 2020, 11:59 (mid-day)**. *The guidelines about late assignments are explained in the students' handbook.*