School of Computing and Information Systems
The University of Melbourne
COMP90042
WEB SEARCH AND TEXT ANALYSIS (Semester 1, 2019)

Workshop exercises: Week 2

Discussion

1. Give some examples of text processing applications that you use on a daily basis.
2. What is **tokenisation** and why is it important?
   (a) What are **stemming** and **lemmatisation**, and how are they different? Give examples from the `WSTA_N1_preprocessing` iPython notebook.
3. Compare using a **term–document matrix** vs. an **inverted index** for resolving a ranked query efficiently.
4. Using the TF-IDF vector space model, using raw term frequency $tf_{d,t}$, $\log \frac{N}{df_t}$ as the inverse document frequency formulation, find the ranking for the query `apple ibm`, based on calculated over the following collection. You should use cosine similarity, but for this question, we will skip the document normalisation step (for time reasons.)

	`apple`	`ibm`	`lemon`	`sun`
D_1	4	0	1	1
D_2	5	0	5	0
D_3	2	5	0	0
D_4	1	0	1	7
D_5	0	1	3	0

5. Recall the Okapi BM25 term weighting formula:

$$w_t = \log \frac{N - f_t + 0.5}{f_t + 0.5} \times \frac{(k_1 + 1) f_{d,t}}{k_1((1-b) + b\frac{L_d}{L_{\text{avg}}}) + f_{d,t}} \times \frac{(k_3 + 1) f_{q,t}}{k_3 + f_{q,t}}$$

   where f_t is the document frequency of term t, $f_{d,t}$ is the term frequency of term t in document d and $f_{q,t}$ is the term frequency of term t in query q. k_1, k_3 and b are parameters with $0 \leq k_1 < \infty$, $0 \leq k_3 < \infty$ and $0 \leq b \leq 1$. L_d is the length of document d and L_{avg} is the average document length in the collection.

   What are its parameters, and what do they signify? How do the components relate to TF (term frequency) and inverse document frequency (IDF)?

Programming

1. Make sure that you have a Python environment where you can run the given iPython notebooks. In particular, ensure that the `numpy`, `sklearn` and `nltk` packages are installed (i.e. you can `import` them).
2. Adapt the `WSTA_N1_preprocessing` iPython notebook into a program which tokenises a input file based on the five–step model given in the lectures.

3. Issue some queries using the small IR engine given in the iPython notebook `WSTA_N2_information_retrieval`. Read (some of) the documents that are returned: confirm that the keyword(s) is/are present, and judge whether you think these documents are relevant to your query.

Catch-up

- Revise the following terms, as they are used in a text processing context: "corpus"; "document"; "term"; "token".
- Revise "stop words", and why they are often removed from a text in a text processing/information retrieval context. Use the Web to find a list of stop words for English — are there any words in the list that you might consider not to be a stop word? Are there any words that you consider to be stop words that are missing from the list?
- Recall the most common regular expression operators; practice writing some regular expressions to solve common text processing problems.
- Remind yourself of the difference between the various evaulation metrics discussed in the lectures this week (**accuracy**, **precision**, **recall**, **f-score**). Re-read the (supervised) machine learning pipeline.
- What is an **information retrieval engine**?
- What does it mean for a document to be **relevant** to a query?
- What is a **vector space model**? How can we find **similarity** in a vector space?
- (Re-)familiarise yourself with Python, if you haven't used it recently. In particular, focus on string and array processing, including regular expressions. Also revise functions and mapping mathematical formulae to Python syntax (including the `numpy` package).
- Familiarise yourself with the Natural Language Toolkit (NLTK). You might like to use the e-book `http://nltk.org/book` as a resource; it also covers some of the basics of Python, if you've never used the language before.

Get ahead

- (Extension) Identify some tokenisation issues in a language (other than English) of your choice. How much alteration would need to be made to the tokenisation strategy from the lectures to account for these issues?
- What effect do the various preprocessing regimes have on the efficiency (time) and effectiveness (relevant results) of querying with the system (note: not building the index)? In particular, consider:
  1. Stemming
  2. Stopping
  3. Tokenisation (e.g. of non-alphabetic tokens)