HW Assignment 2 (Due date: Fri, Oct 25, by 10:30am)

1 Text Processing [100 points]

In this exercise, you will use the indexing model you build for the first homework assignment in order to evaluate the following ranking models:

1. **TF.IDF**: use the standard *tf.idf* weighting scheme *ltc.ltn* in the Vector Space Model approach to IR.
2. **BIM**: The Binary Independence Model approach to IR, where the probabilities p_i of term occurrence in relevant documents are estimated usign each of the following methods:
   (a) **Croft and Harper**: $p_i = 0.5$.
   (b) **Greiff**: $p_i = 1/3 + 2/3 * df_i/N$, where N is the number of documents in the collection.
   (c) **Pseudo-relevance feedback**: Let *VR* be the average number of relevant documents per query (the floor of). Use the pseudo-relevance feedback algorithm with $V = VR$ and $V = VR$ / *2*.
3. **BM25**: use the Okapi BM25 extension to BIM, with parameters $k_1 = 1.5$, $k_2 = 1.5$, and $b = 0.75$.

Evaluate each of the ranking models above on two collections: `CACM` and `Cranfield`. They are both linked from the class web site. Cranfield is a collection of 1400 documents on the topic of aerodynamics. Use text that has been fully preprocessed: tokenization, stemming, stopwords removal, and case folding.

Compute and report the Mean Average Precision (IIR 8.4) for each of the 64 queries in CACM and for the first 64 queries in Cranfield and use it to compute the average MAP for each collection. Compare the performance of the above measures for each collection and across the two collections. Elaborate on differences and on how you think results could be improved.

2 Web Crawler [100 points]

Implement a Web crawler that collects the URLs of webpages from the `ohio.edu` domain. Your crawler will have to perform the following tasks:

1. Start with http://www.ohio.edu.
2. Perform a Web traversal using a breadth-first strategy. You should only crawl HTML pages.
3. Keep track of the traversed URLs, making sure that (a) they are part of the OU domain, (b) they are normalized, and (c) they were not already traversed.

4. Stop when you reach 3000 URLs and save them into a file.

For each document that you crawled, create the following files:

- A file with all the in-links. Each `ohio.edu` link should be marked with a special symbol.
- A file with all the out-links. Each `ohio.edu` link should be marked with a special symbol.
- A file with all the anchor text from the corresponding in-links.

Make sure that you crawler is **polite** i.e. it obeys the robots exclusion rules and it does not hit the site too often. **Run the crawler on your own machine**. Relevant functionality in Python: class `HTMLParser` in module `HTMLParser`, function `urljoin` in module `urlparse`.

3 Submission

Please turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit a directory that has your working code, data, and images, and a concise README file describing these before class. Do not send NLTK code! Create a gzipped, tar ball archive of your directory, and upload it on Blackboard.

For example, if the name is John Williams, creating the archive can be done using the following commands:

> tar cvf williams_john.tar williams_john

> gzip williams_john.tar

These two steps will create the file 'williams_john.tar.gz' that you can upload on Blackboard.

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.
2. Use adequate comments, both block and in-line to document your code.
3. Type and nicely format the project report, including discussion points, tables, graphs etc. so that it is presentable and easy to read.
4. Working code and/or correct answers is only one part of the assignment. The project report, including discussion of the specific issues which the assignment asks about, is also a very important part of the assignment. Take the time and space to make an adequate and clear project report.