CS380 Information Retrieval and Web Search

Lab 3

Grouping and topics for Group Project 1

Monday, Feb 10

To aid in these projects, I have installed (or will soon have installed) python NLTK (Natural Language ToolKit). For this set of projects you can use that resource. If you are using a different programming language and come across something with the a similar set of capabilities, it is probably OK to use. Check with me.

The topics listed below are sketches rather than full-blown statements. These sketches should be sufficient to do topic selection. Next week in lab it is my intention to give time for each group to meet and flesh out a full idea and then discuss that idea with me.

Project topics:

1. Stemming. Find copies of public domain stemmers (The Porter stemmer and other are available as part of python NLTK.) Apply these to combined corpus for your group. How much compression? Find examples of mistakes in each stemmer. Read and implement the method described in "Corpus-Based Stemming using Co-Occurrence of word variants" by Xu and Croft. Analyze along the lines of Croft and Xu on your corpus. Finally, get the corpus of another group. Repeat tests. Do conclusions still hold?

2. Term Weighting. Read "Term Weighting Approaches in Automatic text retrieval" by Salton and Buckley. Consider this deeply. Find an alternative weighting method — for instance Signal/Nose Ratio or "Computation of Term Document Discrimination values by use of the cover coefficient concept" by Can and Ozkarahan. (There are many others.) Use this alternate weighting in your your IR system. Come up with a standard set of queries. Judge recall precision of top 3 for both TFIDF and you other weighting scheme.

3. Implement an inverted index with full positional information. Write queries that illustrate its use. Discuss the changes in storage requirements and retrieval processing. You will also need a way of allowing users to specify

phrases and phrases with wildcards (ie, you should be able to handle a query like "'from' within 5 words of 'Trump').

4. Build a full boolean query engine. It need only support AND and OR. (It would be cool if it also supported NOT.) Discuss complexities of boolean query processing. Do everything efficiently!
5. Build a B-tree based inverted index. Write queries that illustrate its use. Does google/bing appear to use B-trees? Can you tell? Literature review for yea-nay. Handle incremental document additions.
6. Bi-grams and trigrams. The google dataset has sufficient information to calculate cohesion or Mutual Information for the bigrams and trigrams. Do this calculation for the entire set. Analyze your results in terms of what are the most interesting — to your eye — bi-grams and trigrams identified as important by any of these stats. Propose and implement some other formula for word Ngram analysis. (Note, many of the files are compressed. Your program must be able to uncompress and read the stream rather than changing the file on disk.)
7. A topic for your suggestion (approved by me) from among the subjects covered to date. (For instance, something about part of speech tagging and how you could integrate it into the indexing process might be interesting. Alternately, implement and discuss the difficulties in implementing a distributed index. Or, use the unix tool `lex` to drive indexing.)

Topic allocation procedure:

Form groups of 2-3 people. 3 is preferred, but also a maximum.

Within your group, decide on your topic preferences. If you choose 7, you must suggest the topic you want to consider.

When groups are formed have topic preferences, we will have a large discussion (not led by me) in which topics will be allocated such that no topic has more than 2 groups.