CSE454 Project Part4: Dealer's Choice
Assigned: Monday, November 28, 2005
Due: 10:30 AM, Thursday, December 15, 2005

# 1 Project Description

For the last part of your project, you should choose what to do. We'll list some options, but these are just suggestions. Grades are a function of how ambitious you are, how well you pull off your objective, the quality of your evaluation, the quality of your write-up, and the size of your team.

For the final part, you may work in groups of one to four. Groups of two or three are recommended. You may continue your current group, reform a previous group, or create a new group.

# 2 Search Mining

As the number of pages on the Web increases over time, it becomes more and more difficult to return relevant, useful, and important results as top-ranked items, no matter how good the ranking function is. This is particularly true in response to short queries.

One set of techniques that could potentially help are the data mining techniques that we studied in class. In particular,

- Text categorization: by placing pages into categories, search engines can group web pages in various ways or change how they rank the pages. For example, see the search engine available at Microsoft.com and how it breaks its results into categories.
- Clustering: by automatically generating groups of pages, a search engine can help a user rapidly navigate through the results or rapidly refine her query. For example, see how the meta-search engine at www.clusty.com uses clustering to group results.

One option for your final project phase is to use a data mining technique to produce a refined "new and improved" version of Nutch search results.

## 2.1 Project Objectives

You have about three weeks to learn about data mining algorithms and other alternate techniques for presenting search results. You may also need to focus on different indexing techniques so that your program executes reasonably quickly.

This project is longer than previous ones (hence the potential of larger groups), and will involve more programming. You will have much more flexibility to define your own project. You may use the existing support code infrastructure, or you may modify it as your project demands.

Don't feel constrained by the existing support code or by common search engines. You can throw out the standard sorted list and present results however you like (e.g., topic-driven tables?

hyperbolic trees?). Or you can demand that users give background info before using the engine so that the clustering is personalized. The only requirements are that your project perform some form of textual data mining to improve search, and that it be of sufficient scope.

Some suggestions:

- Instead of writing your own data mining program, download one from the web. Options include:
  - You might first look at WEKA (http://www.cs.waikato.ac.nz/ml/) and the Automatic Knowledge Miner (http://www.auknomi.com/) for multiple algorithms including Naive Bayes.
  - RAINBOW also contains multiple algorithms (http://www-2.cs.cmu.edu/ mccallum/bow/).
  - Text clustering (www.download.fm/index.php/Reference/Knowledge_Management/Knowledge_Discovery/Text_Mining/),
  - SVM (http://kiew.cs.uni-dortmund.de:8001/mlnet/instances/7f000002e417fa42cd),
  - Slipper (http://www-2.cs.cmu.edu/w̃cohen/slipper/),
  - C4.5 (http://www.cse.unsw.edu.au/q̃uinlan/).

  Choose the program you find easiest to work with. You will find that the actual algorithm is less important than other design choices in the system and most importantly how you represent the problem to the algorithm. Even with prewritten learning code, you will still have lots to do.

- Apply your data mining algorithm "post-retrieval". That is, instead of applying the algorithm to every page in Nutch's index, apply it only to the results of the query. (Perhaps in the post-processing phase.)

- How will your system perform at query-time? What can you precompute to make sure that execution is as fast as possible?

- You should probably not find it necessary to download any pages directly, but can rather find content through the large crawl that was made available in the last assignment. If you do decide to download pages, do so in parallel with a short timeout otherwise the process will take too long. Also, be sure your download code is "polite."

- What is the vector representation of a web page? Is it "bag of words"? What words should you exclude from the vector? Should you take word frequencies into account? How about TF/IDF?

- Examine the CSE454 support source code. You can take and modify this code for your own project if you so choose. Or, you can use it as it stands.

## 3 Non-Internet Search

Enterprise search (search through a corporation's Intranet, stored behind the firewall, typically with only employee access) requires very different ranking techniques than general Internet search. For example, there are typically many fewer hyperlinks than in the web as a whole, so page rank is less beneficial. Furthermore, the most commonly linked page is often the least useful (so pagerank actually hurts). Finally, the date of page creation (or last modification) is usually much more important. As a case in point, on Veteran's Day I tried to search the UW website (`www.washington.edu/home/search.html`) and the top hit was a 1998 article about quality of the HUB food! Not what I wanted. So your challenge would be to devise the best ranking scheme possible for a local crawl. (This option depends on our ability to get you a local crawl, so please let us know asap if it is of interest).

Another, related, option would be to search Microsoft documentation in an effort to provide an improved "help" experience. Existing help systems are notoriously bad and even Microsoft employees agree that the best way to get help is to Google for it. One should be able to do much better with a honed, specialized ranking function. If you are interested, we will provide a crawl of a subset of these files (e.g., for just Office help).

## 4 Active Interfaces

if you are getting sick of search, indices and ranking functions, why not work on an improved interface? You could use AJAX or the technology of your choice to provide tool tips, keyword completion (ala Google Suggest) or some other idea of your choosing. For ideas, check out Browster. Or integrate this with a clustering system (either by finding another group doing clustering work or by building on top of `clusty.com`.

## 5 Something Else

Don't feel limited to these suggestions. If you have another idea, talk to Dan or Alan for feedback and a reality check.

## 6 Getting Started

Some suggestions: Start with a 1-2 page "project-plan" that describes (e.g., which data mining algorithm you've chosen), what you plan to do with it, your key design decisions, and the experiments you plan to run. If you are doing the standard (data-mining / clustering) project, it's essential that you download the data mining code and become comfortable with using it on text as part of formulating your proposal.

Please think through the experiments you intend to make **before** starting to write any code. If code with your experiments in mind, it will be much, much easier to do well on this important aspect.

Dan or Alan is happy to review your plan either in office hours, in a privately-scheduled meeting, or via email — whichever is most convenient for you. Indeed, **meeting with Dan or Alan is recommended**.

# 7 Writeup

Be sure that your writeup contains:

- Your goals for the project
- Your system design and algorithmic choices
- How to start and use your project
- Sample screens of typical usage scenarios
- Experiments and results that show how effective your system is, and where it could be improved.
- Conclusions and ideas for future work
- An appendix detailing which people did what parts of the work, and what externally-written code (if any) was used in your project.

## 7.1 Support code

You can find the Java code for all CSE454 support code at `/projects/instr/cse454-05au/assignment4/reference`. Most of these files you've seen before. The new one is `SegmentsReader2.java`, which implemented the wrapper code for assignment 3.

The `cse454` command-line script from the previous assignments did a lot of work in setting classpaths and other variables. You might find it useful to adapt it for your own project. There's a copy of the script at `/projects/instr/cse454-05au/assignment3/bin`.

That script makes reference to the `cse454.jar` library, which you also needed to compile against. That JAR file contains all the support code from the previous assignments. The class files inside the JAR consist of all of the files from `/projects/instr/cse454-05au/assignment3/reference`. You can find a JAR file that contains the Lucene-only files at `/projects/instr/cse454-05au/assignment4/lib/lucene-1.3-rc2-dev.jar`. (Lucene handles all the text-searching for Nutch.)

You may also find the Lucene documentation useful. Check out `http://jakarta.apache.org/lucene/docs/api/index.html`.

Finally, if you have questions about the support code feel free to ask Alan.

## 8 What to Hand In

- By 10:30am on December 15, please hand in a hard-copy of your report as well as an electronic version of both the report and the code. We will supply details on the electronic hand-in process via the class mailing list.

Late submissions for the final project and writeup *will not* be accepted, as it will be the end of the quarter.

We will *not* be doing verbal project presentations this quarter.

## 9 Groups and Collaboration

You will work with up to three other people on this project. Everyone is expected to contribute equally.

Discussions between different groups are allowed, subject to the Gilligan's Island rule and other important directives listed in the class policies.