CS 158: Course Missive

Spring 2013

Administrative

Time and Location: Mondays and Fridays, 3:00pm to 4:00pm at CIT Lubrano.

Staff

You can contact the TAs by sending an email to `cs158tas@cs.brown.edu`.

Who	Name	Office	Hours	cs ID
Professor	Eli Upfal	319	TBA	`eli`
Professor	Tim Kriska	335	TBA	`kraskat`
Grad TA	Ahmad Mahmoody	507	TBA	`ahmad`
Head TA	Matt Mahoney	271	TBA	`mjmahone`
Ugrad TA	David Storch	271	TBA	`dstorch`

Course Description

This course covers traditional material as well as recent advances in information retrieval (IR), the study of indexing, processing, querying of textual data and the use of crowd-sourcing. The focus will be on techniques geared to hypertext documents available on the World Wide Web. Topics include:

- efficient text indexing,
- boolean and vector space retrieval models,
- web crawling, link-based algorithms, and web metadata,
- text/web clustering and classification,
- crowd-sourced labeling,
- scaling of IR algorithms

See Syllabus section below.

Prerequisites

There are no official prerequisites for this course. However, you are expected to have taken the equivalent of one of Brown's introductory sequences, CSCI0150/0160, CSCI0170/0180, or CSCI0190. These courses cover many of the data structures and underlying algorithms that we will assume you have knowledge of.

Resources

- **Textbook:** *"Introduction to Information Retrieval"* by C. Manning, P. Raghavan, and H. Schutze. Cambridge University Press, 2008 – A free version is available online at:

  `http://nlp.stanford.edu/IR-book/`.

- **Course website:** `http://cs.brown.edu/courses/cs158/`

Syllabus

1. **Introduction:** Goals and history of IR. The impact of the web on IR. The role of artificial intelligence (AI) in IR.
2. **Basic IR Models:** Boolean and vector-space retrieval models; ranked retrieval; text-similarity metrics; TF-IDF (term frequency/inverse document frequency) weighting; cosine similarity.
3. **Basic Tokenizing Indexing, and Implementation of Vector-Space Retrieval:** Simple tokenizing, stop-word removal, and stemming; inverted indices; efficient processing with sparse vectors; Java implementation.
4. **Experimental Evaluation of IR:** Performance metrics: recall, precision, and F-measure; Evaluations on benchmark text collections.
5. **Query Operations and Languages:** Relevance feedback; Query expansion; Query languages.
6. **Text Representation:** Word statistics; Zipf's law; Porter stemmer; morphology; index term selection; using thesauri. Metadata and markup languages (SGML, HTML, XML).
7. **Web Search:** Search engines; spidering; metacrawlers; directed spidering; link analysis (e.g. hubs and authorities, Google PageRank); shopping agents.
8. **Text Categorization and Clustering:** Categorization algorithms: naive Bayes; decision trees; and nearest neighbor. Clustering algorithms: agglomerative clustering; k-means; expectation maximization (EM). Applications to information filtering; organization; and relevance feedback.
9. **Recommender Systems:** Collaborative filtering and content-based recommendation of documents and products.
10. **Information Extraction and Integration:** Information Extraction and Integration: Extracting data from text; XML; semantic web; collecting and integrating specialized information on the web.
11. **Crowd-sourcing** Using crowd-sourcing, in particular micro-task platforms, to annotate data, generate labels, and extract features. Quality control techniques for crowd-sourcing.

Projects

There are about 7 programming assignments that can be implemented in any programming language. However, the recommended language for the class is Python: any support code will be written in Python, and past experience has shown Python to be sufficient in terms of both development and processing time. The deadlines for assignments are **firm**, as we will give you a reasonable amount of time to complete it. In **exceptional cases**, you should ask Prof. Upfal directly for an extension, preferably **before the deadline**.

Note, as some of these projects are new, the order and details are subject to modification.

1. **Warm-Up: Anagram Solver** This project is designed as a warm-up. You will create a very basic dictionary indexer, and then you will be able to query your index with potential anagrams: the querier will return which words in the dictionary are anagrams of your query. This project is designed as a measure of whether you are ready for this course. As such, it will be completed by yourself.
2. **Main Project 1: A Basic Search Engine** This is the first of your search engine projects. You will create an inverted index from a collection of web documents, which will require parsing and indexing the collection. Then, you will create a search engine to process one word, free text, phrase and boolean queries. This project will be done with a partner.
3. **Main Project 2: Extending the Search Engine** This is the second of your search engine projects, done with your partner. You will extend your previous search engine so that it supports phrase, wildcard, and wildcard phrase queries.

4. **Side Project: Hadoop** This project's goal is to teach you how to utilize parallel computing power to operate on large sets of data. Specifically, you will be re-implementing your indexer to work quickly on very large data sets.

5. **An Interlude: Classification** This project, once again, will be done with your main project partner. The goal of this project is to use machine learning techniques to classify documents into specific categories.

6. **Fun project: Crowd-sourcing** With your main partner you will design a simple crowd-sourcing pipeline on Amazon Mechanical Turk to improve the search quality.

7. **Main Project 3: The Final Search Engine** The first half of this project will be done with your main project partner, while the second half will be done by yourself. The goal of this project is to enhance your search engine using methods such as PageRank, crowd-sourcing, machine learning and compression.

Evaluation

You will be evaluated both on the correctness of your solution (i.e. it implements the specifications of the project), as well as on your code's efficiency. In addition, each assignment will include a written report on which you will be graded.

Collaboration Policy

You will be expected to collaborate in CSCI1580. Specifically, for most of the projects, we expect you to have a partner. This partner should not change throughout the course of the semester, as your later projects will build upon the work of your earlier ones. While we encourage you to discuss your projects with other groups, you may not look at any other group's code, nor help in writing the answers to any other group's written questions. For assignments that do not allow for a partner, you are considered to be in a single group containing only yourself. Furthermore, to get credit for this course, you must return a signed Collaboration Policy, available with the first project, acknowledging that you agree to these policies.