CS276B

Text Retrieval and Mining
Winter 2005

Lecture 2

Recap: Lecture 1

- Web search basics
- Characteristics of the web and users
- Paid placement
- Search Engine Optimization

Plan for today

- Overview of CS276B this quarter
- Practicum 1: basics for the project
  - Possible project topics
  - Helpful tools you might want to know about

Overview of 276B

- Consider it the "applications" course built on CS276A in Autumn
- Significant project component
  - Less homework/exams
- A research paper appraisal that you conduct
- Application topics that are "current" and that introduce new challenges:
  - Web search/mining
  - Information extraction
  - Recommendation systems
  - XML querying
  - Text mining

Topics: web search

- Initiated in Lecture 1
- Issues in web search
  - Scale
  - Crawling
  - Adversarial search
- Link analysis and derivatives
- Duplicate detection and corpus quality
- Behavioral ranking

Topics: XML search

- The nature of semi-structured data
- Tree models and XML
- Content-oriented XML retrieval
- Query languages and engines

Topics: Information extraction

- Getting semantic information out of textual data
  - Filling the fields of a database record
- E.g., looking at an events web page:
  - What is the name of the event?
  - What date/time is it?
  - How much does it cost to attend
- Other applications: resumes, health data, …
- A limited but practical form of natural language understanding

Topics: Recommendation systems

- Using statistics about the past actions of a group to give advice to an individual
- E.g., Amazon book suggestions or NetFlix movie suggestions
- A matrix problem: but now instead of words and documents, it's users and "documents"
- What kinds of methods are used?
- Why have recommendation systems become a source of jokes on late night TV?
  - How might one build better ones?

Topics: Text mining

- "Text mining" is a cover-all marketing term
- A lot of what we've already talked about is actually the bread and butter of text mining:
  - Text classification, clustering, and retrieval
- But we will focus in on some of the higher-level text applications:
  - Extracting document metadata
  - Topic tracking and new story detection
  - Cross document entity and event coreference
  - Text summarization
  - Question answering

Course grading

- Project: 50%
  - Broken into several incremental deliverables
- Paper appraisal/evaluation: 10%
- Midterm (or slightly-after-midterm): 20%
  - In class, Feb 15
- Two Homeworks: 10% each
  - See course website for schedule

Paper appraisal (10%)

- You are to read and critically appraise a recent research paper which is relevant to your project
  - Students work by themselves, not in groups
- By Jan 27, you must obtain instructor confirmation on the paper you will read
  - Propose a paper no later than Jan 25
- By Feb 10 you must turn in a 3-4 page report on the paper:
  - Summarize the paper
  - Compare it to other work in the area
  - Discuss some interesting issue or some research directions that arise
    - I.e., not just a summary: there should be some value-add

Paper sources

- Look at relevant recent conferences:
  - Often then find papers at CiteSeer/library or homepage!
  - SIGIR: http://www.sigir.org/sigir2004/draft.htm
  - WWW: http://www2004.org/
  - SIGMOD: [SIGMOD 2004 site seemed dead!]
  - ICML: http://www.aicml.cs.ualberta.ca/_banff04/icml/
  - …

Project (50%)

- Opportunity to devote time to a substantial research project
  - Typically a substantive programming project
- Work in teams of 2-3 students
  - Higher expectation on project scope for teams of 3
  - But same expectation on fit and finish from teams of 2

Project (50%)

- Due Jan 11: Project group and project idea
  - Decision on project group
  - Brief description of project area/topic
  - We'll provide initial feedback
- Due Jan 18: Project proposal
  - Should break project execution into three phases – Block 1, Block 2 and Block 3
    - Each phase should have a tangible deliverable
    - Block 1 delivery due Feb 1
    - Block 2 due Feb 17
    - Block 3 (final project report) due Mar 10
- Jan 20/25: Student project presentations

Project 50% - breakdown

- 5% for initial project proposal
  - Scope, timeline, cleanliness of measurements
  - Writeup should state problem being solved, related prior work, approach you propose and what you will measure.
- 7.5% for deliveries each of Blocks 1, 2
- 30% for final delivery of Block 3
  - Must turn in a writeup
  - Components measured will be overall scope, writeup, code quality, fit/finish.
  - Writeup should be ~8 pages

Project 0% requirements

- These pieces won't be graded, but you do need to do them, and they're a great opportunity to get feedback and inform your fellow students.
- Project presentations in class (about 10 mins per group):
  - Jan 20/25: Students present project plans
  - Mar 8/10: Final project presentations

Finding partners

- If you don't have a group yet, try to find people after class today
- Otherwise use the class newsgroup (su.class.cs276b)

How much time should I spend on my project?

- Of course the quality of your work is the most important part, but...
- Since this is 50% of your grade for a 3-unit course, we figure something like 40 hours per person is a reasonable goal.
- The more you leverage existing work, the more time you have for innovation.

Practicum (Part 1 of 2)

Practicum 1: Plan for today

- Project examples
  - MovieThing
  - Tadpole
  - Search engine spam
  - Lexical chains
  - English text compression
- Recommendation systems
- Tools
  - WordNet
  - Google API
  - Amazon Web Services / Alexa
  - Lucene
  - Stanford WebBase
- Next time: more datasets and tools, implementation issues

MovieThing

- My project for CS 276 in Fall 2003
- Web-based movie recommendation system
- Implemented *collaborative filtering*: using the recorded preferences of a group of users to extrapolate an individual's preferences for other items
- Goals:
  - Demonstrate that my collaborative filtering was more effective than simple Amazon recommendations (used Amazon Web Services to perform similarity queries)
  - Identify aspects of users' preference profiles that might merit additional weight in the calculations
    - Personal favorites and least favorites
    - Deviations from popular opinion (e.g. high ratings of Pauly Shore movies)

MovieThing

MovieThing

Tadpole

- Mahabhashyam and Singitham, Fall 2002
- Meta-search engine (searched Google, Altavista and MSN)
- How to aggregate results of individual searches into meta-search results?
- Evaluation of different rank aggregation strategies, comparisons with individual search engines.
- Evaluation dimensions: search time, various precision/recall metrics (based on user-supplied relevance judgments).

Using Semantic Analysis to Classify Search Engine Spam

- Greene and Westbrook, Fall 2002
- Attempted semantic analysis of text within HTML to classify spam ("search engine optimized") vs. non-spam pages
- Analyzed sentence length, stop words, part of speech frequency
- Fetched Altavista results for various queries, trained decision tree

Judging relevance through identification of lexical chains

- Holliman and Ngai, Fall 2002
- Use WordNet to introduce a level of semantic knowledge to querying/browsing
- Builds on "lexical chain" concept from other research: notion that chains of discourse run through documents, consisting of semantically-related words
- Compare this approach to standard vector-space model

English text compression

- Almassian and Sy, Fall 2002
- Used assumptions about patterns in English text to develop lossless compression software:
  - Separator – word – separator – word …
  - 8 bits per character is usually excessive
  - Zipf's Law – use shorter encodings for more frequent words
  - Stem words and record suffixes
- Achieved performance superior to gzip, comparable to bzip2

Project examples: summary

- Leveraging existing theory/data/software is not only acceptable but encouraged, e.g.:
  - Web services
  - WordNet
  - Algorithms and concepts from research papers
  - Etc.
- Most projects: compare performance of several options, or test a new idea against some baseline

Tools and data

- For the rest of the practicum we'll discuss various tools and datasets that you might want to use
- Many of these are already installed in the class directory or elsewhere on AFS
- Ask us before installing your own copy of any large software package
- We will provide access to a server running Tomcat and MySQL for those who want to develop websites and/or databases (more information soon)

Recommendation systems

- Web resources (contain lots of links):
  - http://www.paulperry.net/notes/cf.asp
  - http://jamesthornton.com/cf/
- Data:
  - EachMovie dataset: 73,000 users, 1600 movies, 2.5 million ratings
  - other data?
- Software:
  - Cofi: http://www.nongnu.org/cofi/
  - CoFE: http://eecs.oregonstate.edu/iis/CoFE/

Recommendation systems: other relevant topics

- Efficient implementations
  - Clustering
  - Representation of preferences: non-Euclidean space?
  - Min-hash, locality-sensitive hashing (LSH)
- Social networks?

WordNet

- http://www.cogsci.princeton.edu/~wn/
- Java API available (already installed)
- Useful tool for semantic analysis
- Represents the English lexicon as a graph
- Each node is a "synset" – a set of words with similar meanings
- Nodes are connected by various relations such as hypernym/hyponym (X is a kind of Y), troponym, pertainym, etc.
- Could use for query reformulation, document classification, …

Google API

- http://www.google.com/apis/
- Web service for querying Google from your software
- You can use SOAP/WSDL or the custom Java library that they provide (already installed)
- Limited to 1,000 queries per day per user, so get started early if you're going to use this!
- Three types of request:
  - Search: submit query and params, get results
  - Cache: get Google's latest copy of a page
  - Query spell correction
- Note: within search requests you can use special commands like link, related, intitle, etc.

Amazon Web Services: E-Commerce Service (ECS)

- http://www.amazon.com/gp/aws/landing.html
- Mostly for third-party sellers, so not that appropriate for our purposes
- But information on sales rank, product similarity, etc. might be useful for a project related to recommendation systems
- Also could build some sort of parametric search UI on top of this

Amazon Web Services: Alexa Web Information Service

- Currently in beta, so use at your own risk…
- Limit 10,000 requests per user per day
- Access to data from Alexa's 4 billion-page web crawl and web usage analysis
- Available operations:
  - URL information: popularity, related sites, usage/traffic stats
  - Category browsing: claims to provide access to all Open Directory (www.dmoz.com) data
  - Web search: like a Google query
  - Crawl metadata
  - Web graph structure: e.g. get in-links and out-links for a given page

Lucene

- http://jakarta.apache.org/lucene/docs/index.html
- If you didn't get enough of it in 276A…
- Easy-to-use, efficient Java library for building and querying your own text index
- Could use it to build your own search engine, experiment with different strategies for determining document relevance, …

Stanford WebBase

- http://www-diglib.stanford.edu/~testbed/doc2/WebBase/
- They offer various relatively small web crawls (the largest is about 100 million pages) offering cached pages and link structure data
- Includes specialized crawls such as Stanford and UC-Berkeley
- They provide code for accessing their data
- More on this next week

Run your own web crawl

- Teg Grenager is providing Java code for a functional web crawler
- You can't reasonably hope to accumulate a cache of millions of pages, but you could investigate issues that web crawlers face:
  - What to crawl next?
  - Adverse IR: cloaking, doorway pages, link spamming (see lecture 1)
  - Distributed crawling strategies (more on this in lecture 5)

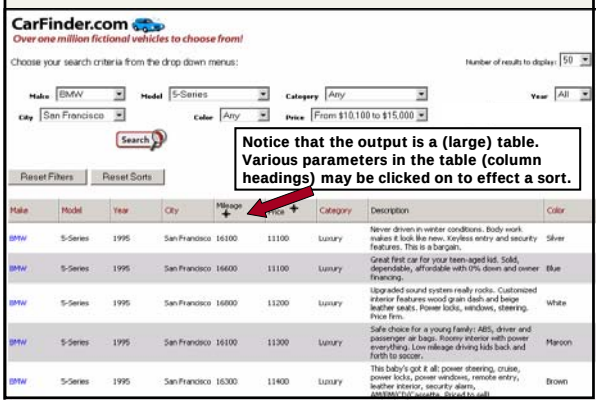
More project ideas

(these slides borrowed from previous editions of the course)

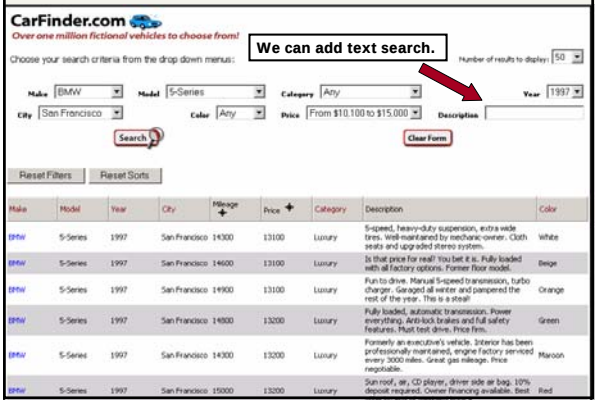
Parametric search

- Each document has, in addition to text, some "meta-data" e.g.,
  - Language = French
  - Format = pdf
  - Subject = Physics etc.
  - Date = Feb 2000
- A parametric search interface allows the user to combine a full-text query with selections on these parameters e.g.,
  - language, date range, etc.

Parametric search example



Parametric search example



Secure search

- Set up a document collection in which each document can be viewed by a subset of users.
- Simulate various users issuing searches, such that only docs they can see appear on the results.
- Document the performance hit in your solution
  - index space
  - retrieval time

"Natural language" search / UI

- Present an interface that invites users to type in queries in natural language
- Find a means of parsing such questions into full-text queries for the engine
- Measure what fraction of users actually make use of the feature
  - Bribe/beg/cajole your friends into participating
  - Suggest information discovery tasks for them
  - Understand some aspect of interface design and its influence on how people search

Link analysis

- Measure various properties of links on the Stanford web
  - what fraction of links are navigational rather than annotative
  - what fraction go outside (to other universities?)
    - (how do you tell automatically?)
- What is the distribution of links in Stanford and how does this compare to the web?
- Are there isolated islands in the Stanford web?

Visual Search Interfaces

- Pick a visual metaphor for displaying search results
  - 2-dimensional space
  - 3-dimensional space
  - Many other possibilities
- Design visualization for formulating and refining queries
- Check www.kartoo.com

Visual Search Interfaces

- Are visual search interfaces more effective?
- On what measure?
  - Time needed to find answer
  - Time needed to specify query
  - User satisfaction
  - Precision/recall

Cross-Language Information Retrieval

- Given: a user is looking for information in a language that is not his/her native language.
- Example: Spanish speaking doctor searching for information in English medical journals.
- Simpler: The user can read the non-native language.
- Harder: no knowledge of non-native language.

Cross-Language Information Retrieval

- Two simple approaches:
  - Use bilingual dictionary to translate query
  - Use simplistic transformation to normalize orthographic differences (coronary/coronario)
- Performance is expected to be worse - By how much?
- Query refinement/modification more important -

  Implications for UI design?

Meta Search Engine

- Send user query to several retrieval systems and present combined results to user.
- Two problems:
  - Translate query to query syntax of each engine
  - Combine results into coherent list
- What is the response time/result quality trade-off? (fast methods may give bad results)
- How to deal with time-out issues?

Meta Search Engine

- Combined web search:
  - Google, Altavista, Overture
- Medical Information
  - Google, Pubmed
- University search
  - Stanford, MIT, CMU
- Research papers
  - Universities, citeseer, e-print archive
- Also: look at metasearch engines such as dogpile, mamma

IR for Biological Data

- Biological data offer a wealth of information retrieval challenges
- Combine textual with sequence similarity
  - Requires BLAST or other sequence homology algorithm
- Term normalization is a big problem (greek letters, roman numerals, name variants, eg, E. coli O157:H7)

IR for Biological Data

- One place to start: www.netaffx.com
  - Sequence data
  - Textual data, describing genes/proteins
  - Links to national center of bioinformatics
- What is the best way to combine textual and non-textual data?
- UI design for mixed queries/results
- Pros/Cons of querying on text only, sequence only, text/sequence combined.

Peer-to-Peer Search

- Build information retrieval system with distributed collections and query engines.
- Advantages: robust (eg, against law enforcement shutdown), fewer update problems, natural for distributed information creation
- Challenges
  - Which nodes to query?
  - Combination of results from different nodes
  - Spam / trust

Personalized Information Retrieval

- Most IR systems give the same answer to every user.
- Relevance is often user dependent:
  - Location
  - Different degrees of prior knowledge
  - Query context (buy a car, rent a car, car enthusiast)
- Questions
  - How can personalization information be represented
  - Privacy concerns
  - Expected utility
  - Cost/benefit tradeoff

Latent Semantic Indexing (LSI)

- LSI represents queries and documents in a "latent semantic space", a transformation of term/word space
- For sparse queries/short documents, LSI representation captures topical/semantic similarity better.
- Based on SVD analysis of term by document matrix.

Latent Semantic Indexing

- Efficiencies of inverted index (for searching and index compression) not available. How can LSI be implemented efficiently?
- Impact on retrieval performance (higher recall, lower precision)
- Latent Semantic Indexing applied to a parallel corpus solves cross-language IR problem. (but need parallel corpus!)

Detecting index spamming

- *I.e., this isn't about the junk you get in your mailbox every day!*
- most ranking IR systems use "frequency of use of words" to determine how good a match a document is
- having lots of terms in an area makes you more likely to have the ones users use
- There's a whole industry selling tips and techniques for getting better search engine rankings from manipulating page content

#3 result on Altavista for "luxury perfume fragrance"

Detecting index spamming

- A couple of years ago, lots of "invisible" text in the background color
- There is less of that now, as search engines check for it as sign of spam

Questions:

- Can one use term weighting strategies to make IR system more resistant to spam?
- Can one detect and filter pages attempting index spamming?
  - E.g. a language model run over pages
- [From the other direction, are there good ways to hide spam so it can't be filtered??]

Investigating performance of term weighting functions

- Researchers have explored range of families of term weighting functions
  - Frequently getting rather more complex than the simple version of tf.idf which we will explain in class
- Investigate some different term weighting functions and how retrieval performance is affected
  - One thing that many methods do badly on is correctly relatively ranking documents of very different lengths
  - This is a ubiquitous web problem, so that might be a good focus

A "real world" term weighting function

- "Okapi BM25 weights" are one of the best known weighting schemes
  - Robertson et al. TREC-3, TREC-4 reports
  - Discovered mostly through trial and error

N is the number of documents in the collection
n_t is the number of documents containing term t
$\mathrm{tf}_{t,d}$ is the frequency of term t in document d
$w_{t,d}$ is the contribution of term t to the relevance of document d

$$w_{t,d} = 0.4 + \frac{0.6 \cdot \mathrm{tf}_{t,d}}{\mathrm{tf}_{t,d} + 0.5 + 1.5\dfrac{\mathrm{length(d)}}{\mathrm{avglen}}} \cdot \frac{\log \dfrac{N + 0.5}{n_t}}{\log N + 1}$$

Investigating performance of term weighting functions

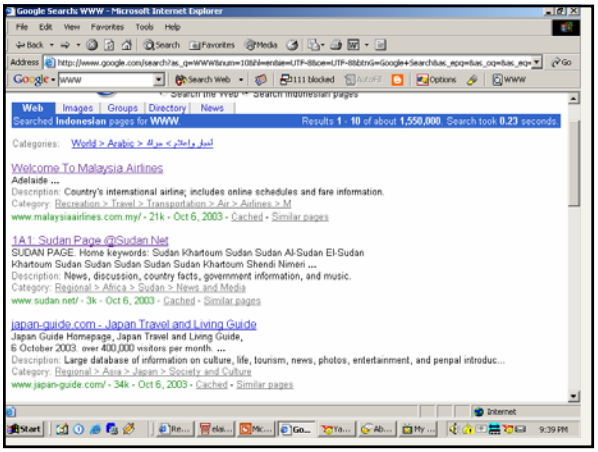
- Using HTML structure:
  - HTML pages have a good deal of structure (sometimes) – in terms of elements like titles, headings etc.
  - Can one incorporate HTML parsing and use of such tags to significantly improve term weighting, and hence retrieval performance?
  - Anchor text, titles, highlighted text, headings etc.
  - Eg: Google

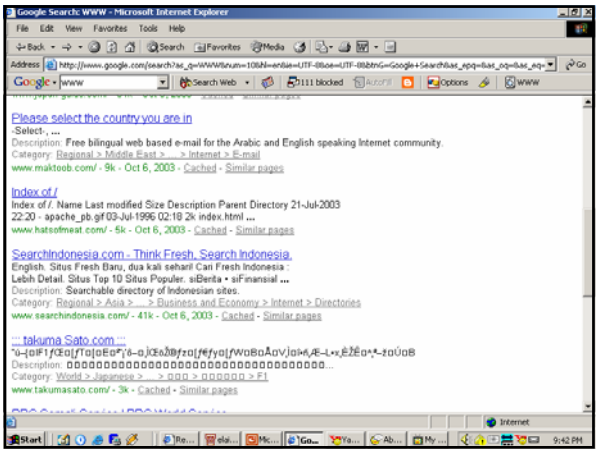
Language identification

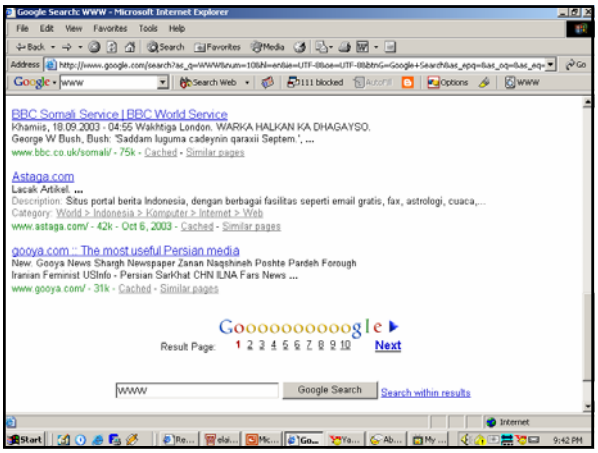
- People commonly want to see pages in languages they can read
- But sometimes words (esp. names) are the same in different languages
- And knowing the language has other uses:
  - For allowing use of segmentation, stemming, query expansion, …
- Write a system that determines the language of a web page

Language identification

- Notes:
  - There may be a character encoding in the head of the document, but you often can't trust it, or it may not uniquely determine the language
  - Character n-gram level or function-word based techniques are often effective
  - Pages may have content in multiple languages
- Google doesn't do this that well for some languages (see Advanced Search page)
  - I searched for pages containing "WWW" [many do, not really a language hint!] in **Indonesian**, and here's what I got…







N-gram Retrieval

- Index on n-grams instead of words
- Robust for very noisy collections (lots of typos, low-quality OCR output)
- Another possible approach to cross-language information retrieval
- Questions
  - Compare to word-based indexing
  - Effect on precision/recall
  - Effect on index size/response time