Information Retrieval with examples from FAST

Lecture in INF5100, Nov 3, 2004
Knut Omang, Dr.scient
1. Aman II / Senior Research Scientist
knuto@ifi.uio.no / knuto@fast.no
www.ifi.uio.no/~knuto / www.fast.no

Overview

• Introduction to Information Retrieval (IR)
• Overview of Information Retrieval (broad def.) techniques
  – IR models (boolean, vector space, probabilistic)
  – Indexing
  – Document analysis
    • Locating and feeding
    • Filtering
    • Classification, similarity
    • Semantic analysis
  – Query analysis
  – Result processing
• Example: FAST Enterprise Search Platform
  – Technology overview
  – Usage examples
  – Perspectives
Managing information

“The challenge is not how to keep useful information, but rather how to discard all the useless...”
- Bjørn Kirkerud, 1993

The art of discarding – ”kjekt å ha”
- Growing number of information sources
- Higher frequency of information updates
  ⇒ avoid drowning in the sea of information..

Data retrieval vs Information retrieval – different entry points – focus points

Data retrieval goal:
- A set of items (formally) satisfying the query

  Missing an item can be fatal, getting an extra item can be fatal!

Information Retrieval goal:
- Find the (most) relevant information satisfying the user’s intent of the query

  If the most relevant result is hit #1046567, the user is likely to miss it!
Information Retrieval

- Old science – draws on/cross-sections into many fields:
  - Library science
  - AI (Artificial Intelligence)
  - Linguistics
  - Statistics
  - Computer Science/databases
  - HCI (Human/Computer interaction)
  - ...

- Useful mental model of IR process:
  Searching for literature in the library

Information Retrieval (def.)

From Wikipedia (www.wikipedia.org):

"Information Retrieval is the art and science of searching for information in documents, searching for documents themselves, searching for metadata which describes documents or searching within databases, whether relational stand alone databases or hypertext networked databases such as the Internet or intranets, for text sound, images or data. There is a common confusion, however, between data, document, information, and text retrieval, and each of these have their own bodies of literature, theory, practice and technologies...."

From "Modern Information Retrieval (Ricardo Baeza-Yates, 1999)":

"Information Retrieval deals with the representation, storage, organization of, and access to information items...."
Retrieval – an iterative search process

“Science is not a mechanism but a human progress, and not a set of findings but a search for them.”
- Jacob Bronowski

Examples:
1. Google search – what would be Google’s goal (apart from making money and having fun...)?
2. Why would IBM use FAST ESP to power IBM.COM?
3. How to you find a document that you downloaded a while ago but don’t know the name of?
4. How do you start your search information about a new topic (lexicon, dictionary, thing to buy, find sci.paper)?

Some answers

1. A search frontend is a success as long as people find what they are looking for
   - Despite that they tend to ask simple and stupid questions...
2. If a search on IBM.com does not (immediately) return the page IBM wants you to see for that query, they lose business to Dell... (who btw also uses Fast ESP ☺)
3. Don’t bother looking for it in the file system – just search for it on the web..
4. Web search in some form ☺...
Example:
- finding a suitable laptop to buy

Traditional database:
select p.name, p.price from product p, productgroup g where g.pnr = p.pnr and g.description like '%laptop%' order by p.price descending

IR-system:
laptop discount

In a database, details of the schema must be known to be able to ask useful questions!

In IR systems there are typically one table – a set of (possibly differently structured, denormalized) documents!

Information Retrieval – model abstraction

Definition (Baeza-Yates 1999)

An information retrieval model is a quadruple \( (D, Q, F, R(q_i, d_j)) \) where

1. \( D \) is a set composed of logical views (or representations) for the documents in the collection.
2. \( Q \) is a set composed of logical views (or representations) for the user information needs. Such representations are called queries.
3. \( F \) is a framework for modeling document representations, queries, and their relationships.
4. \( R(q_i, d_j) \) is a ranking function which associates a real number with a query \( q_i \) in \( Q \) and a document representation \( d_j \) in \( D \). Such ranking defines an ordering among the documents with regard to the query \( q_i \).
A general document representation

- A set of documents  
  \( d_1, \ldots, d_n \)
- The set of terms (keywords) used in (any of) the documents,  
  \( K = \{k_1, \ldots, k_t\} \)
- A weight (significance) of each term in a particular document,  
  \( w_{i,j} = \langle \text{the weight of term } i \text{ in document } j \rangle \)

- Let the vector  
  \( d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j}) \)

  represent the document \( d_j \)

We call \( d_j \) a document vector for the document \( d_j \)

The boolean model

- Set theory + boolean algebra ~ simple SQL way:

  \( w_{i,j} \) is either 0 or 1:

  1 iff \( k_i \) occurs in document \( d_j \)
  0 otherwise

A query is either satisfied or not satisfied for a document
- A document is either relevant or non-relevant

- Advantages
  - Clear formalism behind
  - Simple, well understood, in line with simple database queries

- Disadvantages
  - Often gets too few or too many hits
  - No ranking between hits
The vector model

Allow non-binary weights, \( w_{i,j} \geq 0 \)

- Now the document vector \( d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j}) \) represents a vector in a \( t \)-dimensional space spanned by all the terms (the vocabulary) of the entire document collection.

Document similarity:
- The similarity between documents can be quantified by the correlation of their vectors (for instance \( \cos(u) \) where \( u \) is the angle between the two vectors).

A query \( q \) can be seen as a document with a similar vector representation,

\[ q = (w_{1,q}, w_{2,q}, \ldots, w_{t,q}) \]

Then the similarity between each document and the query "document" can be computed as a rank value.

tf-idf weighting schemes
- common instantiation of vector model

- With \( N \) being the total number of documents in the collection and \( n_i \) being the number of documents in which the term \( k_i \) appears,
  - Let \( freq_{i,j} \) be the frequency (number of occurrences in the text) of term \( k_i \) in \( d_j \).
  - Let \( m_j \) be the term with the highest frequency in \( d_j \).
  - This gives the normalized term frequency \( tf_{i,j} = \frac{freq_{i,j}}{freq_{m,j}} \)
  - The inverse document frequency is given by \( idf_i = \log \frac{N}{n_i} \)
  - Typical weighting function: \( w_{i,j} = tf_{i,j} \times idf_i \)
Simple example (vector model):

Assume *hello* and *world* are the only words that characterizes the docs:

- D1: "A simple hello world test for the world" ($k_m = "world"$, $freq_m,1 = 2$)
- D2: "What a wonderful world this is" ($freq_m,2 = 1$)

Q: *hello world*

The probabilistic model (1)

For a given query there exists a set of documents which contains exactly the relevant documents and nothing else.

- We call this set the *ideal* set.
- Quering an iterative process to find the properties of/query for the ideal document set

1. The user retrieves documents using an approximate query
2. The user selects the most relevant documents in the result set
3. These documents are added to the query
4. Repeat 1-3 until results are good

- Uses binary weights: $w_{ij}$ is either 0 or 1, depending on document is present.
Probabilistic model (2)

- Let $R$ be the documents known (or initially guessed) to be relevant
- Let $\overline{R}$ be the complement of $R$ (the set of not relevant documents)

- Let $P(R|d_j)$ be the probability that the document $d_j$ is relevant to the query $q$ and $P(\overline{R}|d_j)$ the probability that the document $d_j$ is not relevant

  \[
  \text{similarity}(d_j, q) = \frac{P(R | d_j)}{P(\overline{R} | d_j)}
  \]

  - Which by means of statistics (and assuming terms are independent) can be converted to:

  \[
  \text{similarity}(d_j, q) \sim \sum_{i=1}^{w_k} w_{i,q} w_{i,j} \left( \log \frac{P(k_i | R)}{1 - P(k_i | R)} + \log \frac{1 - P(k_i | \overline{R})}{P(k_i | \overline{R})} \right)
  \]

  where $P(k_i | R)$ denotes the probability that the term $k_i$ is present in a document randomly selected from $R$.

Probabilistic model (3)

- Initially, with $R = \emptyset$ (the retrieved documents), set

  \[
  P(k_i, R) = 0.5 \\
  P(k_i, \overline{R}) = n_i / N \quad (N - \text{total # of doc}, n_i \text{ #docs containing the term } k_i)
  \]

- Retrieve documents containing (some of) the query terms ($R$)

  Improving guesses (iterative process):

  - Select a set $V$ containing the $r$ topmost documents from $R$
  - Let $V_j$ be the subset of $V$ containing the term $k_j$
  - Compute new values for $P(k_j, R)$ and $P(k_j, \overline{R})$:

  \[
  P(k_j | R) = \frac{|V_j| + 0.5}{|V| + 1} \quad P(k_j | \overline{R}) = \frac{n_j - |V_j| + 0.5}{N - |V| + 1}
  \]

  - Repeat process..
A model for the relevance of the search result

Recall: To what extent is the relevant results present in the result set (what fraction of all relevant results did we get?)

\[ \text{Recall} = \frac{|\text{Ret} \cap \text{Rel}|}{|\text{Ret}|} \]

Precision: How relevant is the result set? (what fraction of “good” results are there?)

\[ \text{Precision} = \frac{|\text{Ret} \cap \text{Rel}|}{|\text{Rel}|} \]

IR system architecture example: FAST Enterprise Search Platform (ESP)

[Diagram of IR system architecture]

- Scalable Search Engine
- Query Processing
- Result Processing
- Search Client
- Document Management
- Connectors
- Content API
- XML
- Inverted File
- Products
Indexing (classical)

Usually word based, inverted files,

- The document can be viewed as a map from document to keywords (terms)
  \[ d_i \rightarrow \{ k_j \} \]
- The inverted file index is then a map from keywords (terms) to documents:
  \[ k_j \rightarrow \{ d_i \} \]
- Alternative: Character based – any substring searchable
  - Large character sets (UTF8)
- Words not always easy
  - In CJK (Chinese/Japanese/Korean) words are context dependent!
- Performance!!

More advanced indexing(1)

- Substring search (important for CJK)
  - Efficient character based search support
- Proximity support ((word) position indexing) \( \{ p_{n,j,i} \} \) set of positions for each of the \( n \) occurrences of \( k_j \) in \( d_i \).
  \[ k_j \rightarrow \{ d_i, \{ p_{n,j,i} \} \} \]

Example:

Doc: This document is about the Java Language. .... Before I write more about this I need a cup of coffee....

Search: java coffee
Or explicit: java NEAR/2 coffee
More advanced indexing(2)

Efficiently capturing document structure
- XML/HTML,... explicit structure
- Paragraph, sentence,...
- XPath/XQuery support
  http://www.w3.org/TR/xpath, http://www.w3.org/TR/xquery
  Example, try:

Document analysis/processing (1)
- Query independent processing
  - Retrieving content
    - Connectors (databases, files, intranet: permissions?) – push - latency...
    - Crawling (traversing web looking for content...)
  - Normalizing format
    - Character representation (isolatin, US-ascii, ... → UTF8)
    - Html, PDF, Excel, Word, Lotus notes, XYZ,... → “canonical” XML
  - "Noise" filtering
    - Global menus
    - Spam detection
    - Content filtering
  - Analysing/removing markup
    - Add meta-info for each step...
    - Flatten/handle including links
    - Duplicate removal (one doc – many URI’s...)
    - Cross reference analysis (use anchor text information)
Web issues: Some are willing to go far to get ranked

Using link information:

- **Google’s PageRank:**
  - Pages given importance corresponding to what pages link to it
  - Initially: very good – then spam...
  
  *Example:* Scientology church:
  
  scientology.org, scientology.net, scientology.xxx - all linking together!

- **Keyword spamming**
  - Invisible text, meta information
  - Many of the same tricks as mail spam...

- **Content filtering**
  - Dictionaries of words used (ex. pornography, violence)
  - Positive/negative – “you want it – we provide it” ☺
    
    *Ex (serious).*
    
    - Police/advanced crime units...
    - Terror fighting agencies...

- **How big is the Web, really?? (relative to other domains..)**

---

**Crawler**

- The crawler mimics a user clicking away on the internet

- **Central crawler tasks**
  - Retrieve documents
  - Find and follow outbound links from documents
  - Schedule requests according to a set of rules
  - Handle crawler traps

- Sometimes the crawler mimics an
  
  annoying user

  - Deleting content (ex. Wiki...)
  - Chatting/posting on the web
  - Voting on polls
  - Can be quite persistent..

  → importance of robots.txt etc.
Crawler: Challenges

- Uniform Resource Locators (URLs) are not unique
  - Frames
  - Redirects
  - Dynamic content according to user-agent and/or browser capabilities
- JavaScript, flash & applets support
  - Need to execute scripts etc in order to really mimic a user on the web
- Dynamic content
  - Need to handle corner cases like crawler loops
  - “Transient URLs”
    - Cookies
    - Session IDs
  - ~40% of the web changes constantly

Document analysis/processing (2)

- Language detection (possibly multiple...)
- Tokenization (making words out of a string of symbols)
  - Ex. CJK (Chinese/Japanese/Korean)
- Character normalization
  - Normalize accents, case reduction, ß → ss, ...
  - Katakana → Hiragana (Japanese), +++ (+)
  - Application specific....
- Lemmatization (simpler variant: stemming)
  - Ex.(german): buch ←→ bücher, buches, büchern, buchs, buche
  - Exceptions: Proper names, product names
- Phonetic search (ex.names..)
  - Vandefelden, defelden, van der Felden, Felden, Defelden...
- Synonym relations (IBM.com example: laptop → thinkpad)
- Stopwords (the,and,to,be,....)
Stopword examples

Stopwords – poor mans solution:
– Chance of throwing the baby out with the bath water:

*Web* *Images* *Groups* *News* *Franchise* *more »*

Google

“Is” is a very common word and was not included in your search. [details]

• Other examples:
  – “to be or not to be”
  – “CD The the”

• Compare with FAST technology (now Yahoo..):

Document analysis/processing (3)

• Spell checking/correction
  – Levenshtein distance (LD): similarity between two strings, source (s) and target (t). The distance is the number of deletions, insertions, or substitutions required to transform s into t.
    • Examples:
      – parallel – paralel (LD = 1)
      – computer - cmoputer (LD = 1)

• Classification/clustering - determine membership of document in category
  – Fully automatic (unsupervised) or partially manual (supervised)
  – Ex. taxonomy and initial example set + training
  – Typically by use of vector model
Advanced document analysis

- **Entity extraction:**
  - Find and tag "interesting" substrings
    - Abstract terms: "Political science", "entity extraction", "2nd world war"
    - Person/organization names: "Bill Clinton", "Donald Knuth"
    - Places: "New York", "the nordic countries", "Balticum", "Our neighbour’s house"
    - Dates with context: "Bill Clinton was born" → birthdate:
      - Could be indications of the what/who the document sentence is about
      - Could be interesting for some external application...
    - Lots of techniques based on linguistics, heuristics and hints in the text, fantasy is the limit...

- **Sentiment analysis:**
  - Positive/negative attitude towards topic/person/issue, examples:
    - Recent Norwegian newspaper articles about the EU (% positive/negative)
    - Sentiment of analyst statements about a company’s business ($$$$!!)

...Extracting key concepts

... from search results for tennis in Reuters articles:

- **Persons**
  - Slav Grif (70)
  - Martin Hegg (62)
  - Mattias Kjell (62)
  - Peter Sørensen (53)
  - Andrei Agan (27)
  - Boris Bokan (42)
  - Bjørn Borg (19)
  - Aramis Sanchez (19)
  - Stefan Edberg (25)
  - Michael Chervon (29)
  - Roland Garros (22)
  - Amanda Cochrane (23)
  - Mary Pierce (23)
  - Paolo Gat (25)
  - Maria Navratilova (17)
  - Corinna Mohr (21)
  - Thomas Musul (22)
  - Goan Kuang (21)
  - John McEnroe (13)
  - Tim Henman (19)
  - Lindsay Davenport (15)
  - Ivo Minch (15)
  - Tim Gallagher (9)
  - Carlos Moya (13)
  - Richard Krajci (15)
  - Jane Napolitano (16)
  - Jennifer Capriotti (12)
  - Michael Stich (13)
  - Mary Joe (12)

- **Organizations**
  - Tennis Association (15)
  - Tennis Club (10)
  - International Tennis Federation (10)
  - All England Club (7)
  - Sydney International (6)
  - Financial Affairs (7)
  - World Group (1)
  - Association of Tennis Professionals (3)
  - Red Group (1)
  - White Group (1)
  - Lawn Tennis Association (1)
  - Family Federation (2)
  - All England Lawn Tennis Club (1)
  - Arthur Ashe Foundation (1)
  - Owners Hotel (1)
  - Czech Tennis Association (1)
  - Edberg Foundation (1)
  - Park International (1)
  - Divine (1)
  - United States Tennis Association (1)
  - [Other company names] (1)
  - [Other organization names] (1)

- **Topics**
  - World number (6)
  - French Open (7)
  - Grand tour (3)
  - Tennis player (11)
  - Tennis star (23)
  - Banks court (25)
  - Good tennis (20)
  - Open champion (24)
  - Grand Slam champion (21)
  - Grand Slam titles (21)
  - Professional tennis (16)
  - Singles title (19)
  - Tennis centre (17)
  - Australian Open (15)
  - Australian Open champion (17)
  - Grand Slam tournament (17)
  - German tennis (1)
  - Straight sets (10)
  - French Open title (12)
  - Olympic tennis (11)
  - International tennis (13)
  - Tennis tour (10)
  - Athletes (17)
  - Play tennis (13)
  - World tennis (11)
  - Unforced errors (15)
  - World record (11)
  - Youngest Grand Slam champion (10)
  - Service errors (13)
Query analysis:
Capturing the intent of the query

- **User dependent:**
  - Background/interests/knowledge, mode

- **Type of search**
  - Ex. Java coffee or prog.language

- **Relevance may be context dependent, subjective!**

- **User may not be able to formulate a good query – what to do then?**
  - User profile (previous searches, specified interests, language)
  - User mode (geographical location, device, current theme)
  - Query analysis

---

Query analysis (cont.)

**Tools available:** Query rewriting based on:
- Analysis of the query itself
- Other queries using (some of) the same words

**Techniques:**
- Anti-phrasing (handling natural language queries)
  - Filter/de-weight stop words: i want you to give me all you know about cancer ➔ cancer
- Language detection (often not possible)
  - Interesting subsets: Character set in use
- Spell checking/correction
  - May have unfortunate cases: whitehouse ➔ whorehouse (LD = 2)
- Lemmatization
  - Problem: Language dependent!
  - Recall improving... May reduce precision! (same for spell checking)
- Query Segmentation (based on query logs)
  - new york dining ➔ "new york" dining
- Detect category from query
  - _cost less than $5000_ ➔ _price < 5000 AND meta.collection:products_
- Find similar/"give me more hits like this one"/Query by example
- Application of user preferences/profile (generic application specific steps)
Result processing

- **Presentation of results:**
  - Dynamic teaser:
    - text excerpt highlighting best occurrence(s) of the query in the document
  - Dynamic drill-down:
    - present clustering/grouping results for easy refining of the query
  - Ex. FAST-powered [http://www.scopus.com](http://www.scopus.com)

- **Amazon "Other people bying this also bought ..."**
  - Relate to other peoples queries and associated actions:
  - Utilize query/result statistics (from "click-through")

Overview

- **Introduction to Information Retrieval (IR)**
- **Overview of Information Retrieval (broad def.) techniques**
  - IR models (boolean, vector space, probabilistic)
  - Indexing
  - Document analysis
    - Locating and feeding
    - Filtering
    - Classification, similarity
    - Semantic analysis
  - Query analysis
  - Result processing
- **Example: FAST Enterprise Search Platform**
  - Technology overview
  - Usage examples
  - Perspectives
FAST ➔ scalability!

1) Search

**Search Row:** Replicate data for query performance and availability

**Search Column:** Partition data. Scale for volume by adding new columns

Scalability (2) Feeding - Distributed Crawler architecture:
Scalability/fault tolerance (3): Document processing/analysis:

Scalable document processing

Fault tolerant indexing

From distributed Content feeders

Low latency

Real-time alerts: Search in future information...

SEARCH ENGINE TECHNOLOGY
Search existing information

FILTER ENGINE TECHNOLOGY
Monitor future information

Now
Usages:
- **Subscription (ex.Reuters)**
  - customer profiles as (complex) filter queries
- **Mobile phone alerts**
  - stock info, location enabled: traffic jam
- **Alert on similar**
  - Get me more like this (web hit)!

**Note:** Crucial to offer high precision!

---

Fast ESP – platform → Configurable!

Remaining FAST topics:
- Some FAST customer examples
  - Real software – real users – worldwide...
- Some of the ongoing activities
- Some ”unlearning” from this course 😊
  - Predictions...
“Enterprises tell us that their primary goal in investing in new content technologies today is to unify access to all of their information. FAST ESP creates a broad platform for finding and analyzing any kind of information, in any repository. It’s a leap into the future. IDC believes that this product ushers in a new generation of information access platforms that will provide a unified view of enterprise information. It creates real time access to both data and content. That’s impressive.”

- Susan Feldman, IDC’s Research VP for content technologies