Introduction to Text Retrieval

About Myself

- **Zechao Li:**
  - 李泽超
  - Professor

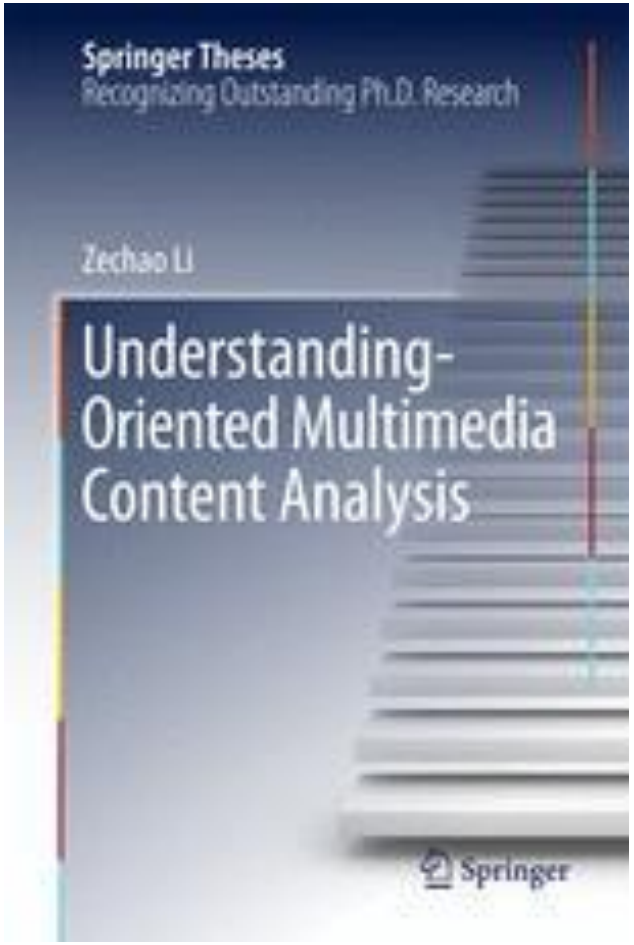
- **My Information:**
  - **Email:** zechao.li@njust.edu.cn
  - **Office:** Room 2061, School of Computer Science and Engineering

  - Education
  - 2004-2008, University of Science and Technology of China
  - 2008-2013, Institute of Automation, Chinese Academy of Sciences
  - 2013-Date, Nanjing University of Science and Technology

  - 智能媒体分析实验室 http://imag.njust.edu.cn/

References:

- **MainText**:
  - Zechao Li. **Understanding-Oriented Multimedia Content Analysis.** Springer 2017



  - https://link.springer.com/content/pdf/10.1007%2F978-981-10-3689-7.pdf

Contents

- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

References:

- Salton (1988), Automatic Text Processing, Addison Wesley, Reading.
  -- for general reading
- Salton G (1972). Dynamic document processing. Comm of ACM, 17(7), 658-668 .
  -- summary of V.S. model
- Price R., Chua Tat-Seng, Al-Hawamdeh S. (1992). Applying Relevance Feedback to a Photo Archival System. Journal of Information Science, 18: 203-215.
  -- Document-Space RF and application in image search

What is Free Text?

- Unstructured sequence of text units with uncontrolled set of vocabulary, Example:

> To obtain more accuracy in search, additional information might be needed - such as the adjacency and frequency information. It may be useful to specify that two words must appear next to each other, and in proper word order
>
> Can be implemented by enhancing the inverted file with location information.

- Information must be analyzed and indexed for retrieval purposes
- Differs from DBMS, which contains structured records:

Name: <s>	Sex: <s>	Age: <i>	NRIC: <s>	

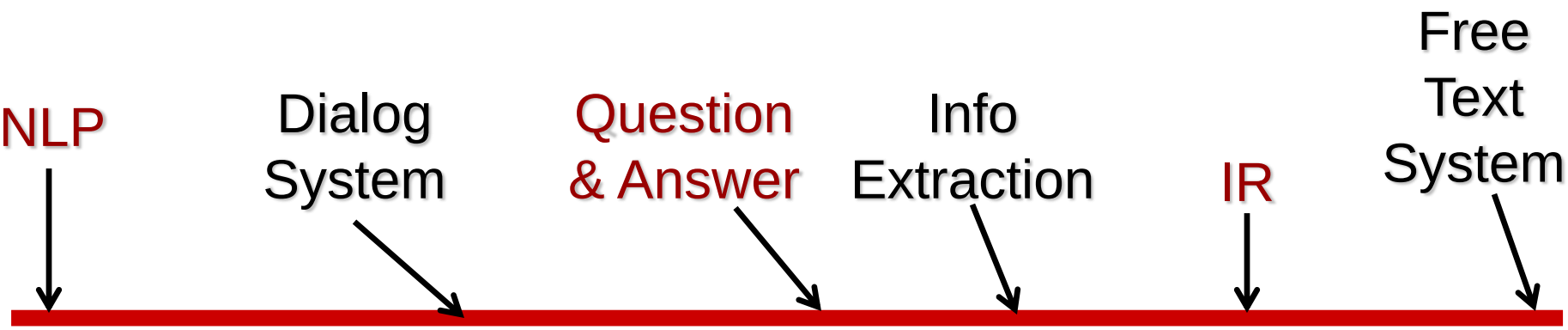
What is Semi-Structured Text?

- Useful information often contains structure
- For example: web page on hotel information:
  - Contains structured info such as <name>, <address>, <price>

<name> Hotel Phoenix </name>
Internet Reservation Office in Singapore:
Tel: <tel> (65) 235-2498 </tel>
Fax: <fax> (65) 235-1416, 235-7620 </fax>
<address> 77 Orchard Road, Singapore 238858 </address>
Located in the heart of Orchard Road, next to Somerset MRT station. Within short walking distance to all major shopping centres and restaurants.

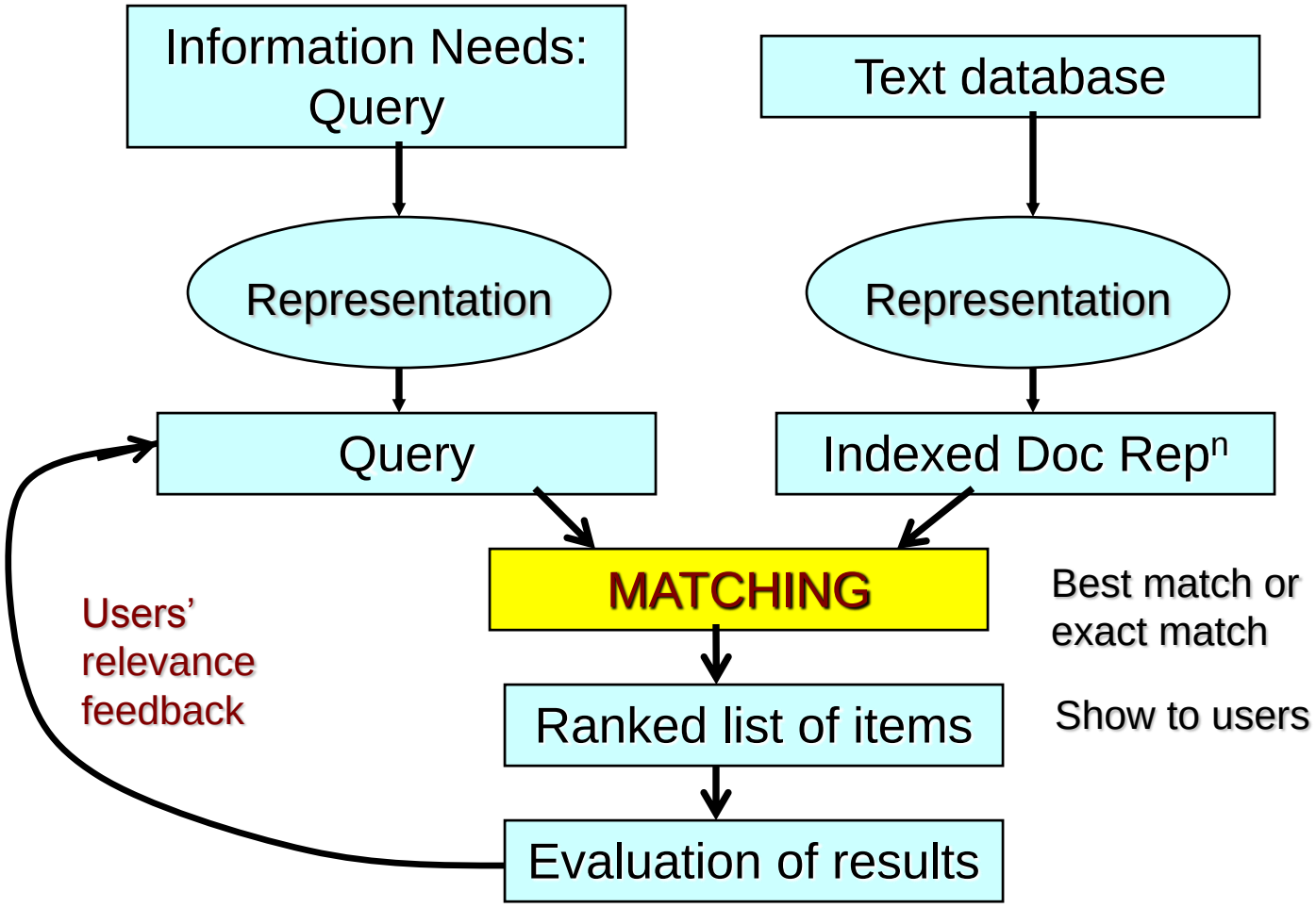
- Definition of structure depends on applications
  - need to separate structured data from un-structured data
  - Basis for XML encoding –advantages of both worlds!
  - Needs to apply Info Extraction technique to extract structured info – mid-level representation (useful for both text and multimedia!)

Types of Text Processing Systems



NLP	Dialog System	Question & Answer	Info Extraction	IR	Free Text System
Understanding content	interactive; Analyze Discourse; Deeper understanding	Provide precise answers to question; Shadow understanding	Extract useful info units and relations, Shadow understanding	Match term occurrence patterns, Little understanding	Pure string matching

Overview of a Typical IR System

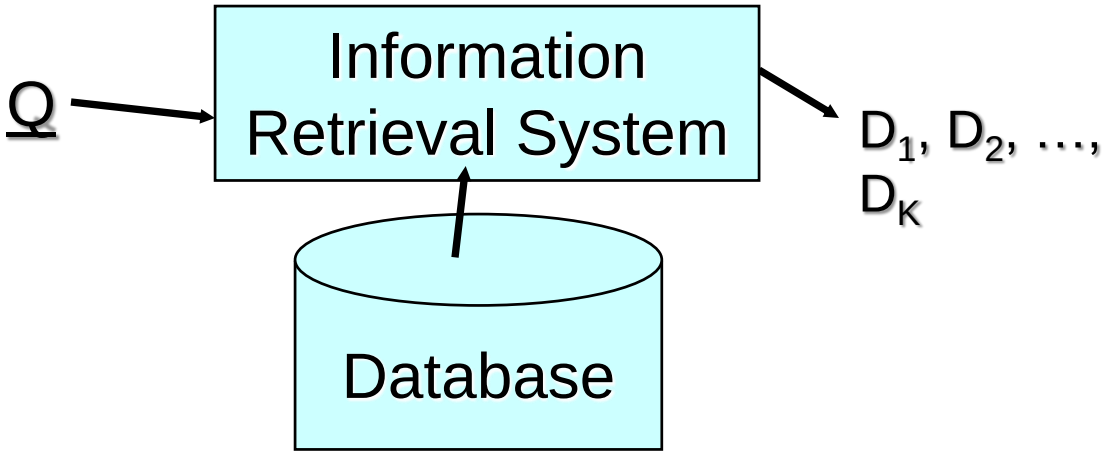


Key Concepts in IR

- Query *Q*: a representation of what the user is looking for
  - can be a list of words or a phrase
- Document: an info entity that the user wants to retrieve
  - can be an actual document of the text annotation of an image
- Collection (or Database): a set of documents
- Index: a representation of info that makes query easier
- Term: word or concept that appears in document or a query

Web Search

- The problem: Given a query Q, and a text collection D, return a ranked list of answers



- We often need to perform a combination of content, link and usage analysis

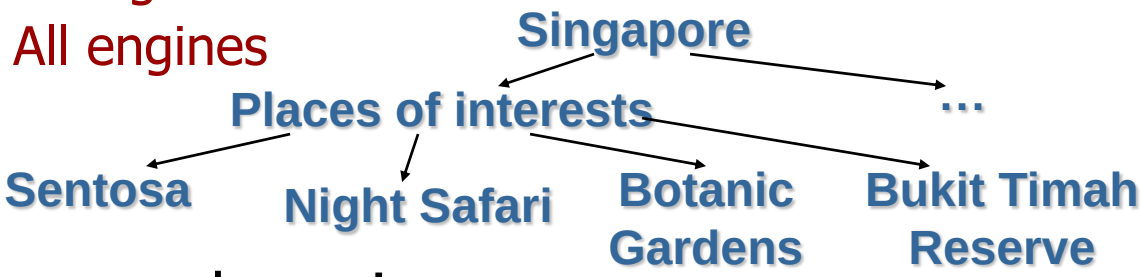
Queries in Real World

- Sample Query Sessions (from AOL)
  - toley spies grames
    tolley spies games
    totally spies games
  - tajmahal restaurant brooklyn ny
    taj mahal restaurant brooklyn ny
    taj mahal restaurant brooklyn ny 11209

    (M: /data4/corpora/AOL-user-ct-collection)
- Characteristics of User Queries
  - Sessions: users revisit their queries.
  - Very short queries: typically 2 words long.
  - A large number of typos.
  - A small number of popular queries. A long tail of infrequent ones.
  - Almost no use of advanced query operators with the exception of double quotes

Application to Internet Search Engines

- Current generation of Internet search engines
  - Based on free-text IR technology
  - Index all non-trivial words in pages into free-text database
  - Users issue free-text query or Boolean (AND, OR) queries
  - Accuracy limited
  - Example of Native search engines: Google, Bing, Baidu …
- Use various means to improve accuracies
  - Use link structures – e.g. Google
  - Use domain concepts – e.g. Yahoo
  - Use query log – e.g. All engines



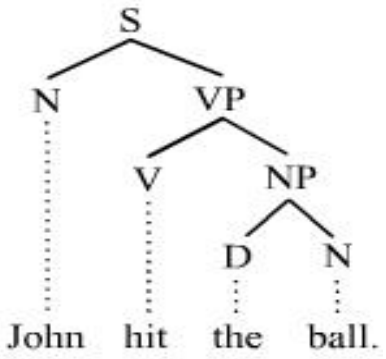
- There are many Meta search engines
  - Uses native search engines to obtain results, re-rank results using various strategies or assume domain knowledge, e.g. Answers.com

Contents

- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

Analysis of Free-Text in IR

- Analyze document, *D*, to extract patterns to represent *D*:
- General problem:
  - To extract minimum set of (distinct) features to represent contents of document
  - To distinguish a particular document from the rest
- Commonly used text features
  - STRING (current commercial systems)
  - Single WORDS (current statistical IR)
  - Info Units: Named entities (IE systems)
  - Linguistic units (NLP)



- Current IR systems are term-based:
  - IR: perform pattern matching, no semantics, general

Example of Text Analysis

<name> Hotel Phoenix </name>
Internet Reservation Office in Singapore:
Tel: <tel> (65) 6235-2498 </tel>
Fax: <fax> (65) 6235-1416, 6235-7620 </fax>
<address> 77 Orchard Road, Singapore 238858 </address>
Located in the heart of orchard road, next to Sommerset MRT station. Within short walking distance to all major shopping centres and restaurants.

- Information Units:
  - IR: term-based: hotel x 1; Singapore x 2; phoenix x 1; internet x 1 ..
  - IE: info units: hotel phoenix; singapore; orchard road; sommerset MRT station; (65)6235-3498; etc and their relations
  - QA: Q: which is the nearest MRT station to Hotel Phoenix?
    A: Sommerset MRT Station
    (Note: if we have good IE, some QA is relatively easy)
  - NLP: Try to understand contents

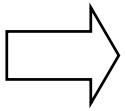
Analysis Example

- Simple representation (single terms) performs quite well

Free-text page:

To obtain more accuracy in search, additional information might be needed - such as the adjacency and frequency information. It may be useful to specify that two words must appear next to each other, and in proper word order.

Can be implemented by enhancing the inverted file with location information.



Text pattern extracted:

information x 3
Words
Word
Accuracy
Search
Adjacency
Frequency
Inverted
File
Location
implemented
….

Term Selection -1

- Research suggests that (INTUITIVE !!):
  - high frequency terms are not discriminating
  - low to medium frequency terms are useful (enhance precision)
- A Practical Term Selection Scheme:
  - eliminate high frequency words
    (by means of a stop-list with 100-200 words)
  - use remaining terms for indexing
- Example, a query: *"I would like documents on expert intermediary systems for online bibliographic searching"*
  would be reduced to
  "*expert intermediary systems online bibliographic searching*"

Stop Word List

- One possible short list (more in the web):

also am an and are be because been

could did do does from

had hardly has have having he hence her here hereby herein hereof hereon hereto herewith him his however

if into it its me nor of on onto or our really

said she should so some such

than thus to too

unto us very

was we were what when where whereby wherein whether which who whom whose why would

you

- http://www.ranks.nl/stopwords

Term Selection -2

- Precision is better served by features that occur frequently in a small number of documents
- One such measure is the Inverse Doc Frequency (idf):

$$\log_2\left(\frac{N}{n_k}\right)+1$$

  - N - total # of doc in the collection
  - n_k - # of doc to which term k is assigned
  - n_k/N gives the prob. of term k in the collection

- EXAMPLE: In a collection of 1000 documents:
  - ALPHA appears in 100 Doc - Wt = 4.322
  - BETA appears in 500 Doc - Wt = 2.000
  - GAMMA appears in 900 Doc - Wt = 1.132

Term Selection -3

- General, idf helps in precision
- tf helps in recall
- Combine both gives the famous tf.idf weighting scheme for a term k in document i as:

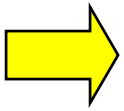
$$w_{ik} = t_{ik} * [\log_2(\frac{N}{n_k}) + 1]$$

Term Selection in Document

Free-text page:

To obtain more accuracy in search, additional information might be needed - such as the adjacency and frequency information. It may be useful to specify that two words must appear adjacent to each other, and in proper word order.

Can be implemented by enhancing the inverted file with location information.



Text pattern extracted:

information x 3
words
word
accuracy
search
Adjacency
adjacent
frequency
inverted
file
location
implemented
….

to x 3
in x 2
the x 3
and x 2
is
more
might
that
such
as
two
by
….

Stop Word List

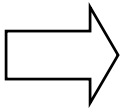
Stemming of Terms -1

- What are the possible problems here?

Free-text page:

To obtain more accuracy in search, additional information might be needed - such as the adjacency and frequency information. It may be useful to specify that two words must appear adjacent to each other, and in proper word order.

Can be implemented by enhancing the inverted file with location information.



Text pattern extracted:

information x 3
Words
Word
Accuracy
Search
Adjacency
adjacent
Frequency
Inverted
File
Location
implemented
….

Stemming of Terms -2

- NEXT PROBLEM:
  - Terms come in different grammatical variants
  - Synonym, Antonym & other word relations
- To solve this problem:
  - Use dictionary such as WordNet
    - resolve synonym, antonym (needs context)
    - reduce terms to "morph" form
  - The other simpler way: apply stemming algorithm
- AIMS of Stemming Algorithm:
  - to reduce the number of words/terms
  - to remove the variants in word forms, such as:
    RECOGNIZE, RECOGNISE,
    RECOGNIZED, RECOGNIZATION
  - hence it helps to identify similar words
  - DEMO: SMILE Stemmer (Wikipedia page on Stemming)

Stemming Algorithm

- Simplest stemming algorithms:
  - only remove suffixes by operating on a dictionary of common word endings, such as -SES, -ATION, -ING etc.
- Simplest is the Porter's algorithm (1980):
  - Iterative:
    - WILLINGNESS → remove NESS, then ING
  - Handle many exception rules:
    - do not remove -ABLE from TABLE, -S from GAS, or -ING from SING
    - eliminate doubling of terminal consonants FORGETTING and FORGET
  - Context-free vs. Context-sensitive
  - Overall steps:
    - 60 suffixes grouped into five steps
    - few context-sensitive and recoding rules
- Points about Stemming:
  - not a linguistic exercise, simply to improve IR performance
  - might alter the meaning of a word

Putting All Together

- Term selection and weighting for Docs:
  - extract unique terms from documents
  - stem terms
  - remove stop words
  - Optionally:
    - *use thesaurus – to group low freq terms*
    - *form phrases – to combine high freq terms*
    - assign, say, tf.idf weights to stems/units
- Do the same for query

Demo of Thesaurus: http://www.merriam-webster.com/

A Simple Similarity Measure

- Represent both query and document as weighted term vectors:
  - $\underline{Q} = (q_1, q_2, \ldots q_M)$
  - $\underline{D_i} = (d_{i1}, d_{i2}, \ldots d_{iM})$ M: # of terms
- A possible query-document similarity is:
  - $sim(\underline{Q},\underline{D_i}) = \Sigma(q_j . d_{ij}), \quad j = 1,.. M$

  may be normalized → cosine similarity formula

A Retrieval Example

- Given:

$\underline{Q}$ = "information", "retrieval"

$\underline{D}_1$ = "information retrieved by VS retrieval methods"

$\underline{D}_2$ = "information theory forms the basis for probabilistic methods"

$\underline{D}_3$ = "He retrieved his money from the safe"

- Document representation (M=10, N=3):

{info, retriev, method, theory, VS, form, basis, probabili, money, safe}

$\underline{Q}$ = {1, 1, 0, 0 …}

$\underline{D}_1$ = {1, 2, 1, 0, 1, …}

$\underline{D}_2$ = {1, 0, 1, 1, 0, 1, 1, 1, 0, 0}

$\underline{D}_3$ = {0, 1, 0, 0, 0, 0, 0, 0, 1, 1}

- The results:
  - Use the correlation formula: sim $(\underline{Q},\underline{D}_i) = \sum(q_j \cdot D_{ij})$
  - sim $(\underline{Q},\underline{D}_1) = 3$; sim $(\underline{Q},\underline{D}_2) = 1$; sim $(\underline{Q},\underline{D}_3) = 1$
  - Hence $\underline{D}_1 >> \underline{D}_2$ and $\underline{D}_3$

Contents

- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

Document representations

- Term-document matrix (m x n)
- Document-document matrix (n x n)
- Typical example in a medium-sized collection: 3,000,000 documents (n) with 50,000 terms (m)
- Typical example on the Web: n=30,000,000,000, m=1,000,000

  n >> m
- Boolean vs. integer-valued matrices

Storage issues

- Imagine a medium-sized collection with n=3,000,000 and m=50,000
- How large a term-document matrix will be needed?
- Is there any way to do better? Any heuristic?

- Instead of an incidence vector, use a posting table:
  - Information: D_1, D_2
  - Retrieval: D_1, D_3
- Use linked lists to be able to insert new document postings in order and to remove existing postings.
- Keep everything sorted! This gives you a logarithmic improvement in access.

Inverted Index Structure -1

- Documents are stored in inverted index

Direct index:

File 1	vector space model
File 2	vector analysis model
File 3	vector algebra and analysis
File 4	space analysis

Inverted index:

Compute n_i, and pre-compute idf

vector	File1, File2, File3
space	File1, File4
model	File1, File2
analysis	File2, File3, File4
algebra	File3

- Given the query: "vector", "space"
  - Needs to access all files in direct index, requires N accesses
    Can be very expensive if N is large (order of billions)
  - For inverted index, needs only 2 file accesses
    Independent of N

Inverted Index Structure -2

- Inverted file accesses:
  - Vector → $File_{vector}$: {File1, File2, File3}
  - Space → $File_{space}$: {File1, File4}
  - $File_{vector} \prod File_{space}$ → {File1}
  - What is the number of operations needed??
- Advantages of Inverted File Structure
  - Most Web queries are short – only need to access few query term indices, rather than all the files

- Inverted file may contain more information:
  - Such as positions of words in document, paragraph etc..
  - What can you do with it??

Basic operations on inverted indexes

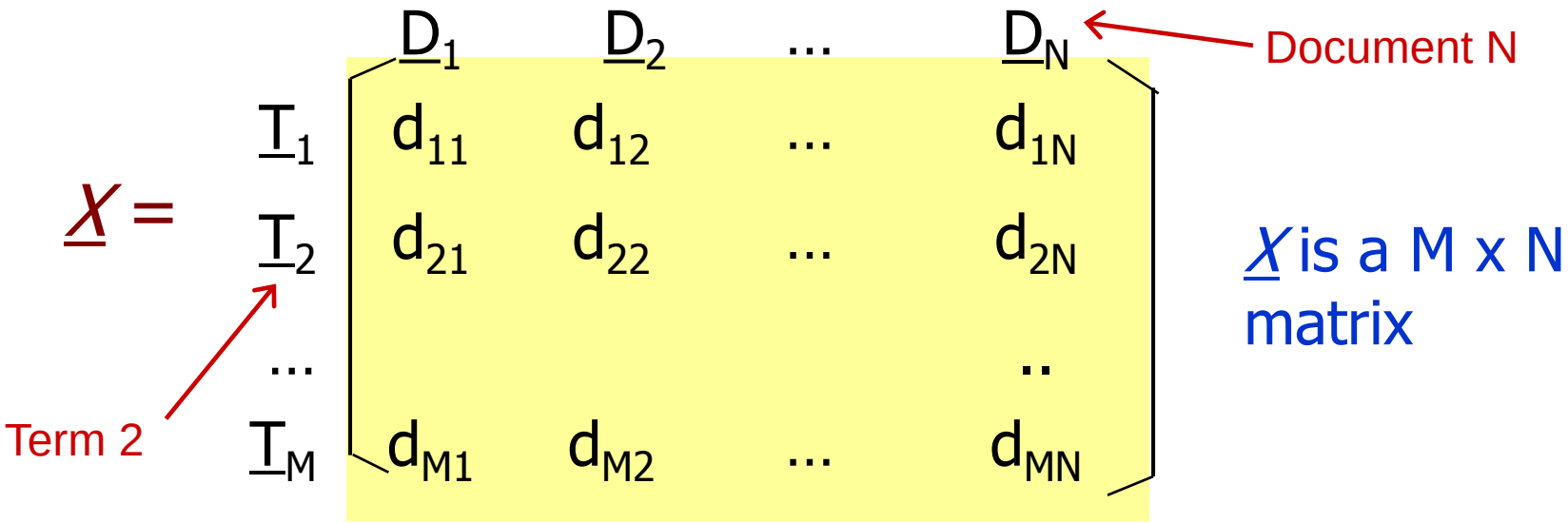
- Conjunction (AND) – iterative merge of the two postings: O(x+y)
- Disjunction (OR) – very similar
- Negation (NOT) – can we still do it in O(x+y)?
  - Example: Vector AND NOT Space
  - Example: Vector OR NOT Space
- Recursive operations
- Optimization: start with the smallest sets

Contents

- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

Document-Term Matrix

- **Document Matrix:** for N documents with M terms, we get:



- **Query Vector:** Q = (q_1, q_2, ... q_M)

Similarity Measures -1

(http://www.scholarpedia.org/article/Similarity_measures)

- Matching is done between a document and a query (or between two documents) in document space
- Distance vs. similarity measures
  - Euclidean distance, City-block distance, Word overlap, Jaccard coefficient, etc
- Euclidean distance: $$Dist(\underline{D}_i, \underline{Q}) = \sqrt{\sum_{k=1}^{N} (d_{ik} \bullet q_k)^2}$$
- City-block distance: $$CityB(\underline{D}_i, \underline{Q}) = \sqrt{\sum_{k=1}^{N} | d_{ik} \bullet q_k |}$$
- Remarks:
  - Both are difference measures
  - Euclidean distance is valid when D & Q dimensions are perceptually integral, whereas city-block is more appropriate when they are perceptually separable (e.g. color and shape, or color and text)

Similarity Measures -2

(http://www.scholarpedia.org/article/Similarity_measures)

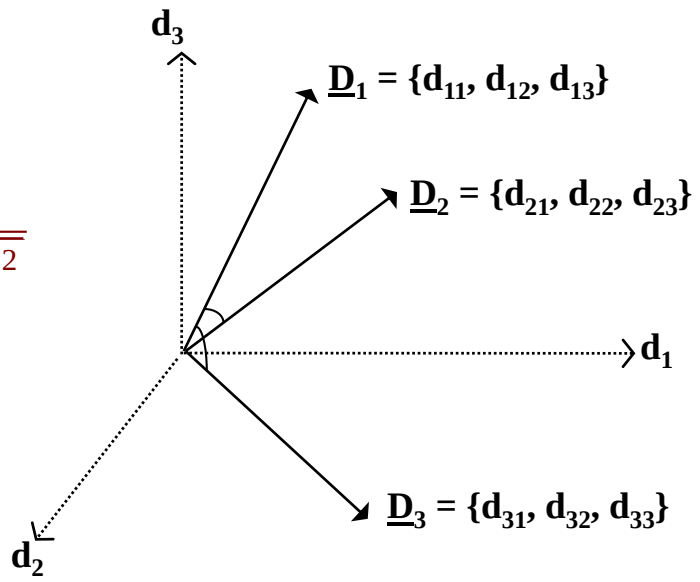
- The Jaccard coefficient: $Jac(\underline{D}_i, \underline{Q}) = \frac{|\underline{D}_i \cap Q|}{|\underline{D}_i \cup Q|}$
- Term correlation: $Corr(\underline{D}_i, \underline{Q}) = \sum (d_{ik} * q_k)$
- Cosine Similarity: (normalization of Correlation) $CosSim(\underline{D}_i, \underline{Q}) = \frac{\sum (d_{ik} * q_k)}{\sqrt{\sum (d_{ik})^2 * \sum (q_k)^2}}$

- Remarks:
  - They are all similarity measures
  - Jaccard is more for sets, whereas Cosine Similarity is more for vectors of continuous variables; both are normalized measures.
  - Cosine is widely used in high dimensional positive space; it is efficient to evaluate especially for sparse vector
  - We normally prefer a similarity rather than difference formula. WHY?

Graphical Interpretation of
Cosine Similarity Measure

- Graphically, it is represented as an equal length vector in t-dimensional space

$$CosSim(Di, Q) = \frac{\sum(d_{ik} * q_k)}{\sqrt{\sum(d_{ik})^2 * \sum(q_k)^2}}$$



* CosSim measures the angle between unit length vectors

- Retrieval of documents can be made to depend on a particular threshold (say 0.5) or a specific number of documents to be retrieved (say 10)

Vector Space Model

- The Model
  - Developed by Salton as part of SMART Retrieval Systems
  - Probably the most popular, widely used and studied model
- Basic Assumptions
  - a fixed sized term set is used to represent documents & queries (non-efficient representation)
  - terms are un-related (orthogonal) to each other
- The vector representation:
  - $\underline{Q} = (q_1, q_2, \ldots q_M)$
  - $\underline{D_i} = (d_{i1}, d_{i2}, \ldots d_{iM})$

  where q_k and d_{ik} are either binary or weighted (using tf.idf)

Vector-Space Model

Weighting of Terms in Documents & Queries

- Document term weights are calculated using a variant of tf.idf

$$\frac{d_{ij}}{\max\{d_{ij}\}} * \log(\frac{N}{n_j})$$

Note that tf is normalized

- Query term weight is:

  Use above or simply $\log(\frac{N}{n_j})$ if q_j is present – WHY?

- Similarity is computed using cosine rule
  - Note that idf is used twice in the cosine formula

Vector-Space Model

Early Results

- Results of SMART-MEDLARS comparison
  - (450 doc, 29 queries)
  - MEDLARS - Manual, controlled indexing
  - SMART - Automatic indexing

Method	Recall	Precision
MEDLARS	0.3117	0.6110
SMART Word Stem	0.2622 (-16%)	0.4901 (-19%)
SMART Thesaurus	0.3232 (+4%)	0.6106 (0%)

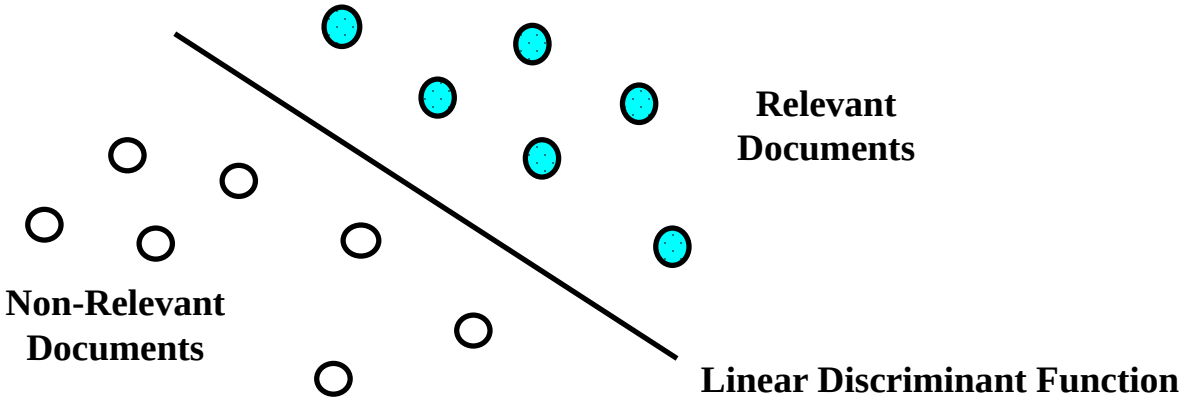
Summary of VS Model

- Advantages:
  - Simplicity
  - able to handle weighted terms
  - easy to modify term vectors
- Disadvantages:
  - term independent assumption
  - First order (does not consider correlation between terms/documents)
  - Synonym and polysemy problems – same for all IR model
- Possible Solutions:
  - Use thesaurus, dictionary etc to overcome snyonym & polysemy problems
  - Apply mathematical analysis to remove term dependency. Eg SVD (Singular Value Decomposition) → LSA

Contents

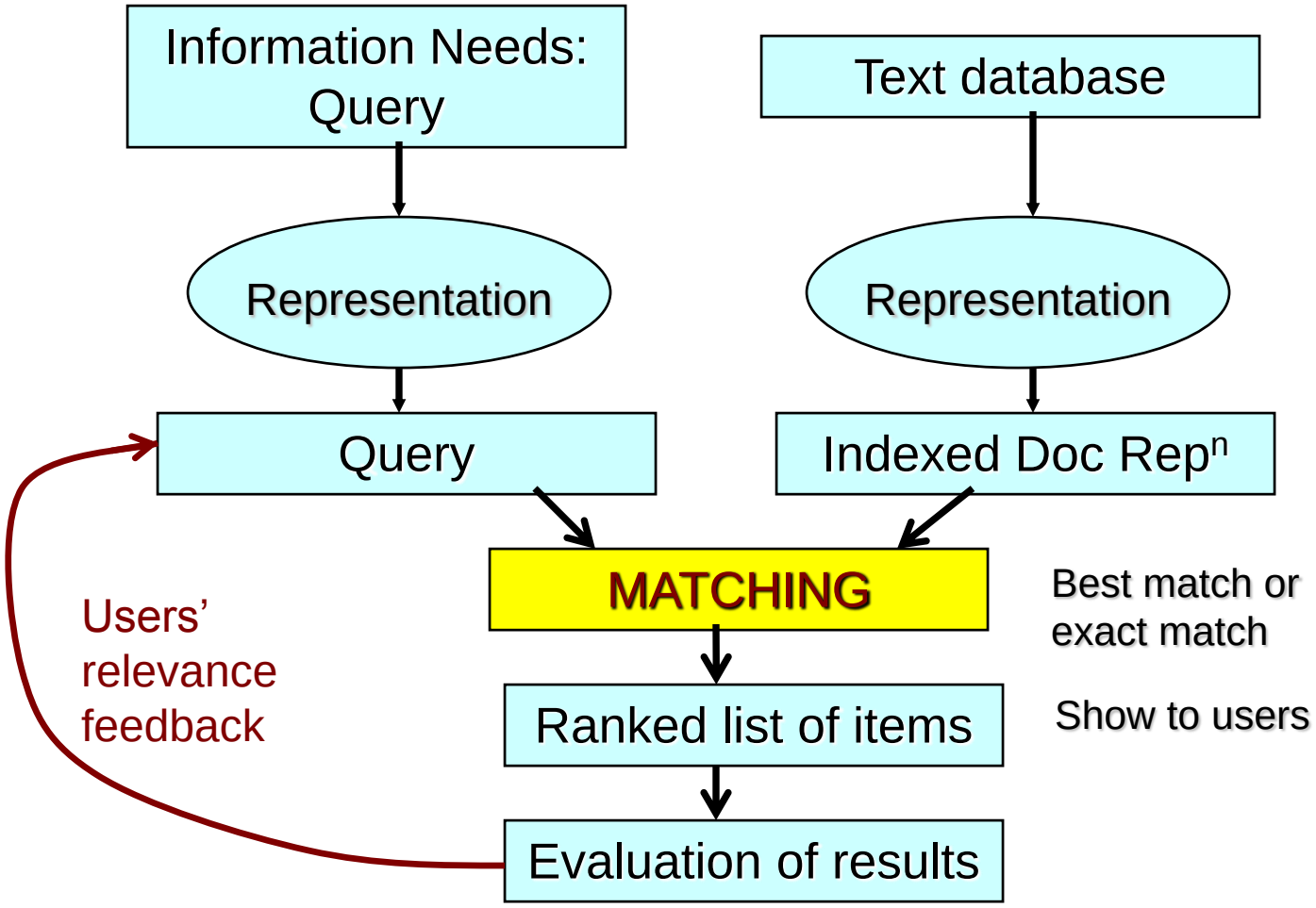
- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

RF Techniques



- Relevance feedback uses some (users') judgments about the relevance of documents to modify weights of discriminant function
- If documents are linearly separable, a learning rule can be specified that will always converge to the optimal discriminant function

Query-Space Relevance Feedback



Query Space RF

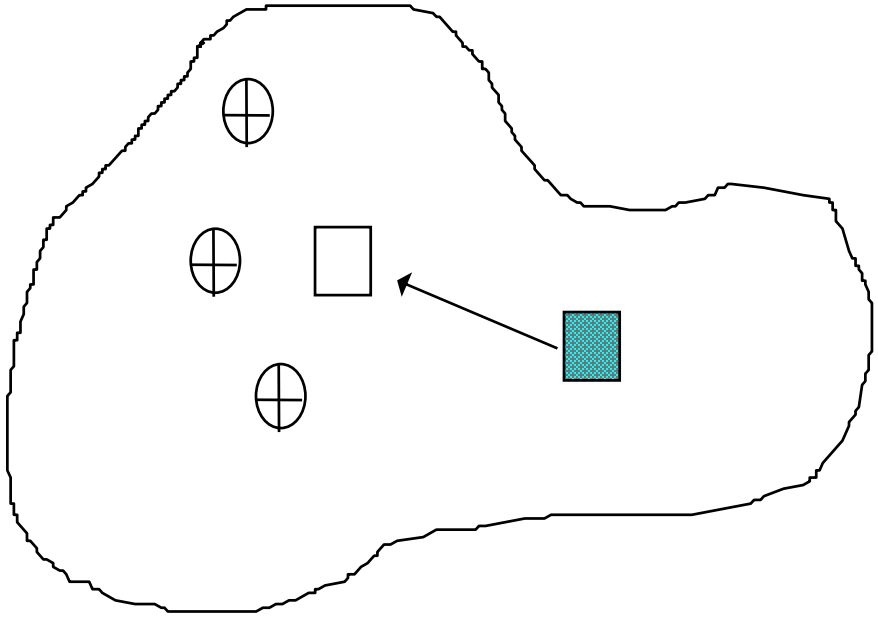
- Main Assumption on Query Space RF:
  - relevant documents resemble each other
  - Query is re-formulated based on known relevant documents
- Query re-formulation is based on two complementary operations
  - terms occur in relevant documents are added to original query vector (or weight of such terms is increased);
  - terms occur in non-relevant documents are deleted from original query vector (or weight of such terms is reduced)

$$\underline{Q}^{(i+1)} = \underline{Q}^{(i)} + \alpha \bullet \sum_{\underline{D}_i \in R} \underline{D}_i - \beta \bullet \sum_{\underline{D}_j \in NR} \underline{D}_j$$

Note: The hard part is in estimating the value for α and β

Query Space RF: Graphical View

- Effects of relevance feedback



Original Query

Feedback Query

Relevant Items Retrieved

$$\sum_{D_i \in N_R} S(\underline{Q}, \underline{D}_i) = \frac{1}{R} \sum_{D_i \in N_R} S(\underline{Q}, \underline{D}_i)$$

- In general, we use only the relevant and top-most non-relevant documents (De-hi method) - WHY?
- In practice, needs to limit size of query too – WHY & HOW to do it?

Query Space RF Process

- R/F Process:
  1. Users issue query, $\underline{Q}$
  2. System returns top N relevant documents, $\{\underline{D_k}\}$, k=1,..N
  3. User provides relevance judgment, $\{\underline{R_k}\}$, k=1,..N_K
  4. Compute weights of terms to be added to query:

     For VS model, rank terms using: $\alpha \sum_{D_i \in R} D_i - \beta \sum_{D_j \in NR} D_j$

  5. Select top n terms, $\{r_i\}$, I=2, .. N
  6. $\underline{Q}^{(i+1)} = \underline{Q}^{(0)} + \{r_i\}$

- There could be problem of query drift
  - How to prevent it?

Results of Query-based RF

- Experimental results: (450 documents, 29 queries)
  - Medlars and SMART comparison

Method	Recall	Precision
Medlars	0.3117	0.6110
SMART Word Stem	0.2622 (-16%)	0.4901 (-19%)
1st Iteration Feedback	0.3235 (+4%)	0.6385 (+5%)
2nd iteration Feedback	0.3433 (+10%)	0.6892 (+13%)

- RF has been found to be extremely effective
  - We expect the first 2-3 iterations to result in over 50% improvement in accuracy and recall.

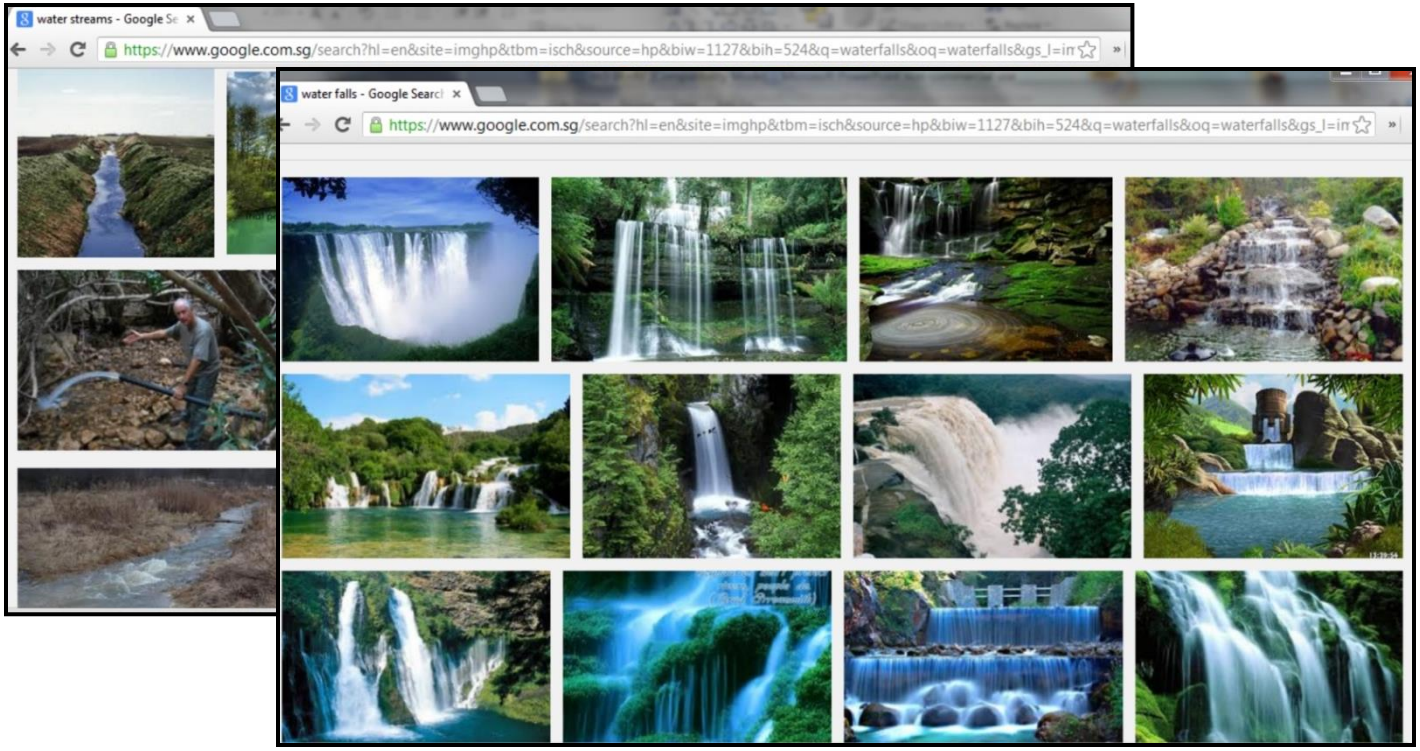
Pseudo relevance feedback

- "Pseudo"-relevance feedback automates the "manual" part of true relevance feedback.
- Pseudo-relevance algorithm:
  - Retrieve a ranked list of hits for the user's query
  - Assume that the top k documents are relevant.
  - Do relevance feedback (e.g., using V.S. RF Scheme)
- Works quite well on average

- But can go horribly wrong for some queries.
- Several iterations can cause query drift – Why and how to prevent it?

Issues in Query-Space RF

- RF is normally very effective:
  - For example: query of "waterfalls before and after RF (simulated)



- Problems with query space RF approach??
  - Benefits only the current query session, but not long term
  - Does not benefit from "wisdom" of other users

Perform Document-Space RF

- Possible solution: perform RF in document space (DRF)
  - modify contents of relevant and/or irrelevant documents to make documents more easily retrievable in future by other users
  - Example, if RF add "waterfall" into IR docs, then future use of term "waterfall" will retrieve these docs
  - Popular in image retrieval to add text annotation to images

- Perform DRF in 2 complementary operations:
  - Query terms are added in relevant documents with initial weight λ, or weight of such terms is increased by δ;
  - Query terms appear in non-relevant documents are decremented by weight σ, and such term is removed if its weight is $< \mu$

    where λ, δ, σ & θ are small constants.

Possible Problems in Document-Space RF

- Documents may lose original meanings
  - Possible solution: need to preserve original terms and limit the number of new terms added (and their weights)??
- Cannot support personalized search
- Cannot support changing user needs:
- Overall, the approach is ad hoc…
  - A better way is to record users' search history in query log, and perform log-based analysis

Contents

- Basic Concepts in Information Retrieval
- Analysis of Text
- Inverted Index
- Similarity Measures
- Relevance Feedback
- Evaluation Measures

(http://en.wikipedia.org/wiki/Information_retrieval)

Evaluation Measures

- Importance of Evaluations
- Efficiency vs. effectiveness
  - Efficiency measured using speed and storage overhead
  - Effectiveness measured using relevance

- For retrieval based on a query, we have:

	Relevant	Non-Relevant
Retrieved	**a**	**b**
Missed	**c**	**d**

N=a+b+c+d ➔ total number of documents DB

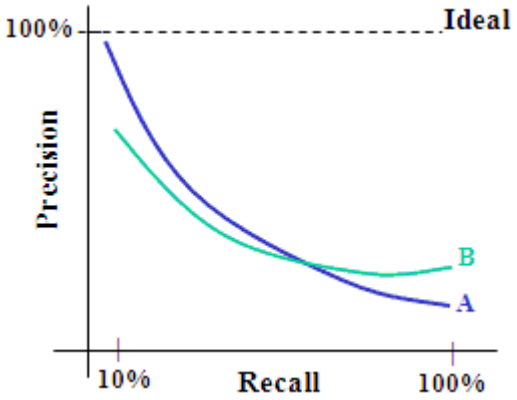
Effectiveness:

- Precision (P) = a / (a+b)
- Recall (R) = a / (a+c)

Evaluation Measures -2

- Results over many collections are compared. However, it is generally more convenient to present a single number

- Present precision value at fixed recall intervals:
  - Define P(R), precision at recall value R
  - Interpolate P(R), for R = 0, 0.1, 0.2, …1.0
  - Present as Recall-Precision graph



  - Or present as a single AveP value

$$\text{AveP} = \frac{1}{11} \sum_{R\epsilon\{0,0.1,0.2,..,1.0\}} P_{\text{interp}}(\text{R})$$

Evaluation Measures -3

- Another common single value measure is:

  $F_\beta = [(\beta^2+1) P R] / [\beta^2 P + R]$

- When both P & R have equal weights, i.e. when $\beta = 1$:

  $F_1 = [2 P R] / [P + R]$

  This is popularly used in retrieval evaluations

Evaluation Measures -4

- In practice, the total # of relevant items is not known, a more popular form of AveP used is:

$$\text{AveP} = \frac{\sum_{k=1}^{n}(P(k) \times \text{rel}(k))}{\text{number of relevant documents}}$$

  - where rel(*k*) =1 if document at rank *k* is relevant; zero otherwise
  - Note that the average is over all relevant documents

- Example: If returned result is (1 means relevant, 0 irrelevant):

  1, 0, 0, 1, 1, 1

  1/1, 0, 0, 2/4, 3/5, 4/6 ←-- precision @ k

  AveP = (1 + 2/4 + 3/5 + 4/6) / 4 = 0.69

- Why not average over all document retrieved??
  - For example, for ranked list of [1, 0, 0, …. 0], what is AveP?

Evaluation Measures -5

- Mean Average Precision (MAP): Average over all queries

$$\mathrm{MAP} = \frac{\sum_{q=1}^{Q} \mathrm{AveP(q)}}{Q}$$

SUMMARY & NEXT LESSON

- This chapter discusses:
  - Introduction to IR
  - Similarity Measures
  - Relevance Feedbacks
  - Evaluation measures
- Next Lesson
  - Image Content Analysis and Retrieval

Additional Slides

Theoretical Basis for VS Model -1

- Vector Algebra
  - t distinct terms available, each term i is identified with a term vector $\underline{T}_i$
  - a vector space is defined whenever the t vectors are linearly independent
  - $\underline{D}_r$ can be written as linear combination of $\underline{T}_i$'s as:

$$\underline{D}_r = \sum(d_{ri}\underline{T}_i)$$

X =

	$\underline{D}_1$	$\underline{D}_2$	...	$\underline{D}_N$
$\underline{T}_1$	d_{11}	d_{12}	...	d_{1N}
$\underline{T}_2$	d_{21}	d_{22}	...	d_{2N}
...				..
$\underline{T}_t$	d_{t1}	d_{t2}	...	d_{tN}

Theoretical Basis for VS Model -2

- Similarity between
  $\underline{D}_r = (\underline{d}_{r1}, d_{r2}, \ldots d_{rt})$, and
  $\underline{Q}_s = (q_{s1}, q_{s2}, \ldots q_{st})$ can be computed as:

$$SIM(\underline{D}_r \cdot \underline{Q}_s) = \sum\sum(d_{ri}\ q_{sj}\ \underline{T}_i \cdot \underline{T}_j) \qquad (1)$$

- Given the term independent assumption:
  - we have $\underline{T}_i.\underline{T}_j = 0$ except when i = j in which case $\underline{T}_i \cdot \underline{T}_j = 1$
  - Thus Equation (1) becomes:

$$SIM(\underline{D}_r \cdot \underline{Q}_s) = \sum\sum(d_{ri}\ q_{sj}) \quad \rightarrow \text{Inner product}$$

- Note that if $\underline{T}_i$ & $\underline{T}_j$ belong to the same thesaurus class, then $\underline{T}_i \cdot \underline{T}_j = 1$

Theoretical Basis for VS Model: Example

Figure 10.3 Sample query-document similarity computations (from [3]). (a) Sample documents and query. (b) Assumed term correlations. (c) Similarity computations for uncorrelated terms. (d) Similarity computations for correlated terms.

$$D_1 = 2T_1 + 3T_2 + 5T_3$$
$$D_2 = 3T_1 + 7T_2 + 1T_3$$
$$Q = 0T_1 + 0T_2 + 2T_3$$

(a)

	T_1	T_2	T_3
T_1	1	0.5	0
T_2	0.5	1	−0.2
T_3	0	−0.2	1

(b)

$$\text{sim}(D_1,Q) = 2\cdot 0 + 3\cdot 0 + 5\cdot 2 = 10$$
$$\text{sim}(D_2,Q) = 3\cdot 0 + 7\cdot 0 + 1\cdot 2 = 2$$

(c)

$$\begin{aligned}\text{sim}(D_1,Q) &= (2T_1+3T_2+5T_3)\cdot(2T_3)\\ &= 4T_1\cdot T_3 + 6T_2\cdot T_3 + 10T_3\cdot T_3\\ &= -6\cdot 0.2 + 10\cdot 1\\ &= 8.8\end{aligned}$$

$$\begin{aligned}\text{sim}(D_2,Q) &= (3T_1+7T_2+1T_3)\cdot(2T_3)\\ &= 6T_1\cdot T_3 + 14T_2\cdot T_3 + 2T_3\cdot T_3\\ &= -14\cdot 0.2 + 2\cdot 1\\ &= -0.8\end{aligned}$$

(d)