

Text Based Search Engines

Qiang Yang
Simon Fraser University

Thanks: Professor Dik Lee, HKUST

4/4/01 Text-based SE 1

Search Engine Topics

- Keyword Based Search Engines
 - Document analysis: Stemming, Stop Words,
 - Ranking: TF-IDF, Vector Space Model
 - Relevance Feedback: interactive search engines
 - Evaluation: precision, recall, others
- Link based search engines: Google, Hubs and Authorities
 - www.searchenginewatch.com
 - www.searchengines.com

4/4/01 Text-based SE 2

Keyword Extraction

- Goal:
 - given N documents, each consisting of words, t_i
 - extract the most significant subset of words → keywords
 - Example
 - [All the students are taking exams] --> [student, take, exam]
- Keyword Extraction Process
 - remove stop words
 - stem remaining terms
 - collapse terms using thesaurus
 - build inverted index
 - extract key words - build key word index
 - extract key phrases - build key phrase index

4/4/01 Text-based SE 3

Stop Words and Stemming

- From a given Stop Word List
 - [a, about, again, are, the, to, of, ...]
 - Remove them from the documents
- Or, determine stop words
 - Given a large enough corpus of common English
 - Sort the list of words in decreasing order of their occurrence frequency in the corpus
 - Zipf's law: Frequency * rank ≈ constant
 - most frequent words tend to be short
 - most frequent 20% of words account for 60% of usage

4/4/01 Text-based SE 4

Zipf's Law -- An illustration

Rank(R)	Term	Frequency (F)	R*F (10**6)
1	the	69,971	0.070
2	of	36,411	0.073
3	and	28,852	0.086
4	to	26,149	0.104
5	a	23,237	0.116
6	in	21,341	0.128
7	that	10,595	0.074
8	is	10,009	0.081
9	was	9,816	0.088
10	he	9,543	0.095

4/4/01 Text-based SE 5

Resolving Power of Word

4/4/01 Text-based SE 6

Simple Indexing Scheme Based on Zipf's Law

Use term frequency information only:

- ① Compute frequency of term k in document i , $Freq_{ik}$
- ② Determine total collection frequency
 $TotalFreq_k = \sum Freq_{ik}$ for $i = 1, 2, \dots, n$
- ③ Arrange terms in order of collection frequency
- ④ Set thresholds - eliminate high and low frequency terms
- ⑤ Use remaining terms as index terms

4/4/01

Text-based SE

7

Stemming

- The next task is stemming: transforming words to root form
 - Computing, Computer, Computation → compute
- Suffix based methods
 - Remove "ability" from "computability"
 - "..."+ness, "..."+ive, → remove
- Suffix list + context rules

4/4/01

Text-based SE

8

Thesaurus Rules

- A thesaurus aims at
 - classification of words in a language
 - for a word, it gives related terms which are broader than, narrower than, same as (synonyms) and opposed to (antonyms) of the given word (other kinds of relationships may exist, e.g., composed of)
- Static Thesaurus Tables
 - [anneal, strain], [antenna, receiver], ...
 - Roget's thesaurus
 - WordNet at Princeton

4/4/01

Text-based SE

9

Thesaurus Rules can also be Learned

- From a search engine query log
 - After typing queries, browse...
 - If query1 and query2 leads to the same document
 - Then, Similar(query1, query2)
 - If query1 leads to Document with title keyword K,
 - Then, Similar(query1, K)
 - Then, transitivity...
- Microsoft Research China's work in WWW10 (Wen, et al.) on Encarta online

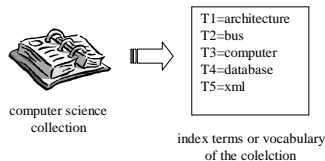
4/4/01

Text-based SE

10

The Vector-Space Model

- T distinct terms are available; call them *index terms* or the *vocabulary*
- The index terms represent important terms for an application → a vector to represent the document
 - $\langle T_1, T_2, T_3, T_4, T_5 \rangle$ or $\langle W(T_1), W(T_2), W(T_3), W(T_4), W(T_5) \rangle$



4/4/01

Text-based SE

11

The Vector-Space Model

- Assumptions: words are uncorrelated

Given:

1. N documents and a Query
2. Query considered a document too
2. Each represented by t terms

3. Each term j in document i has weight d_{ij}

4. We will deal with how to compute the weights later

$$\begin{pmatrix}
 T_1 & T_2 & \dots & T_t \\
 D_1 & d_{11} & d_{12} & \dots & d_{1t} \\
 D_2 & d_{21} & d_{22} & \dots & d_{2t} \\
 \vdots & \vdots & \vdots & \dots & \vdots \\
 D_n & d_{n1} & d_{n2} & \dots & d_{nt}
 \end{pmatrix}$$

$Q \quad q_1 \quad q_2 \quad \dots \quad q_t$

4/4/01

Text-based SE

12

Graphic Representation

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

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

• Is D_1 or D_2 more similar to Q ?
 • How to measure the degree of similarity? Distance? Angle? Projection?

4/4/01 Text-based SE 13

Similarity Measure - Inner Product

- Similarity between documents D_i and query Q can be computed as the inner vector product:

$$\text{sim}(D_i, Q) = \sum_{k=1}^t (D_i \cdot Q) = \sum_{j=1}^t d_{ij} * q_j$$
- Binary: weight = 1 if word present, 0 o/w
- Non-binary: weight represents degree of similarity
 - Example: TF/IDF we explain later

4/4/01 Text-based SE 14

Inner Product -- Examples

Binary:

Size of vector = size of vocabulary = 7

$D = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ (terms: retrieval, database, architecture, computer, text, management, information)
 $Q = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$

$\rightarrow \text{sim}(D, Q) = 3$

Weighted

$D_1 = 2T_1 + 3T_2 + 5T_3$
 $Q = 0T_1 + 0T_2 + 2T_3$
 $\text{sim}(D_1, Q) = 2*0 + 3*0 + 5*2 = 10$

4/4/01 Text-based SE 15

Properties of Inner Product

- The inner product similarity is unbounded
- Favors long documents
 - long document \Rightarrow a large number of unique terms, each of which may occur many times
 - measures how many terms matched but not how many terms *not* matched

4/4/01 Text-based SE 16

Cosine Similarity Measures

- Cosine similarity measures the cosine of the angle between two vectors
- Inner product normalized by the vector lengths

$$\text{CosSim}(D_i, Q) = \frac{\sum_{k=1}^t (d_{ik} \cdot q_k)}{\sqrt{\sum_{k=1}^t d_{ik}^2} \cdot \sqrt{\sum_{k=1}^t q_k^2}}$$

4/4/01 Text-based SE 17

Cosine Similarity: an Example

$D_1 = 2T_1 + 3T_2 + 5T_3$ $\text{CosSim}(D_1, Q) = 5 / \sqrt{38} = 0.81$
 $D_2 = 3T_1 + 7T_2 + T_3$ $\text{CosSim}(D_2, Q) = 1 / \sqrt{59} = 0.13$
 $Q = 0T_1 + 0T_2 + 2T_3$

D_1 is 6 times better than D_2 using cosine similarity but only 5 times better using inner product

4/4/01 Text-based SE 18

Document and Term Weights

- Document term weights are calculated using frequencies in documents (tf) and in collection (idf)
 - tf_{ij} = frequency of term j in document i
 - df_j = document frequency of term j
 - = number of documents containing term j
 - idf_j = inverse document frequency of term j
 - = $\log_2 (N/df_j)$ (N : number of documents in collection)
- Inverse document frequency -- an indication of term values as a document discriminator.

4/4/01

Text-based SE

19

Term Weight Calculations

- Weight of the j th term in i th document:

$$d_{ij} = tf_{ij} \bullet idf_j = tf_{ij} \bullet \log_2 (N/df_j)$$

- TF \rightarrow Term Frequency
 - A term occurs frequently in the document but rarely in the remaining of the collection has a high weight
 - Let $\max\{tf_{ij}\}$ be the term frequency of the most frequent term in document j
 - Normalization: term frequency = $tf_{ij}/\max\{tf_{ij}\}$

4/4/01

Text-based SE

20

An example of TF

- Document=(A Computer Science Student Uses Computers)
- Vector Model based on keywords (Computer, Engineering, Student)
 - $Tf(\text{Computer}) = 2$
 - $Tf(\text{Engineering}) = 0$
 - $Tf(\text{Student}) = 1$
 - $\text{Max}(Tf) = 2$
 - TF weight for:
 - Computer = $2/2 = 1$
 - Engineering = $0/2 = 0$
 - Student = $1/2 = 0.5$

4/4/01

Text-based SE

21

Inverse Document Frequency

- Df_j gives a the number of times term j appeared among N documents
- $IDF = 1/DF$
- Typically use $\log_2 (N/df_j)$ for IDF
- Example: given 1000 documents, computer appeared in 200 of them,
 - $IDF = \log_2 (1000/200) = \log_2(5)$

4/4/01

Text-based SE

22

TF IDF

- $d_{ij} = (tf_{ij}/\max\{tf_{ij}\}) \bullet idf_j$
 $= (tf_{ij}/\max\{tf_{ij}\}) \bullet \log_2 (N/df_j)$
- Can use this to obtain non-binary weights
- Used in the SMART Information Retrieval System by the late Gerald Salton and MJ McGill, Cornell University to tremendous success, 1983

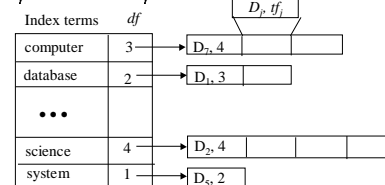
4/4/01

Text-based SE

23

Implementation based on Inverted Files

- In practice, document vectors are not stored directly; an inverted organization provides much better access speed.
- The index file can be implemented as a hash file, a sorted list, or a B-tree



4/4/01

Text-based SE

24

A Simple Search Engine

- Now we have got enough tools to build a simple Search engine (documents == web pages)
 1. Starting from well known web sites, *crawl* to obtain N web pages (for very large N)
 2. Apply stop-word-removal, stemming and thesaurus to select K keywords
 3. Build an inverted index for the K keywords
 4. For any incoming user query Q,
 1. For each document D
 1. Compute the Cosine similarity *score* between Q and document D
 2. Select all documents whose *score* is over a certain threshold T
 3. Let this result set of documents be M
 4. Return M to the user

4/4/01

Text-based SE

25

Remaining Questions

- How to crawl?
- How to evaluate the result
 - Given 3 search engines, which one is better?
 - Is there a quantitative measure?

4/4/01

Text-based SE

26

Measurement

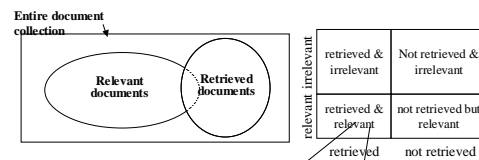
- Let M documents be returned out of a total of N documents;
- $N = N1 + N2$
 - N1 total documents are relevant to query
 - N2 are not
- $M = M1 + M2$
 - M1 found documents are relevant to query
 - M2 are not
- Precision = $M1/M$
- Recall = $M1/N1$

4/4/01

Text-based SE

27

Retrieval Effectiveness - Precision and Recall



$$\text{recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of relevant documents}}$$

$$\text{precision} = \frac{\text{Number of relevant documents retrieved}}{\text{total Number of documents retrieved}}$$

4/4/01

Text-based SE

28

Precision and Recall

- Precision
 - evaluates the correlation of the query to the database
 - an indirect measure of the completeness of indexing algorithm
- Recall
 - the ability of the search to find all of the relevant items in the database
- Among three numbers,
 - only two are always available
 - total number of items retrieved
 - number of relevant items retrieved
 - total number of relevant items is usually not available

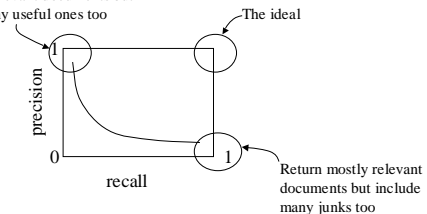
4/4/01

Text-based SE

29

Relationship between Recall and Precision

Return relevant documents but miss many useful ones too



4/4/01

Text-based SE

30

Fallout Rate

- Problems with precision and recall:
 - A query on "Hong Kong" will return most relevant documents but it doesn't tell you how good or how bad the system is!
 - number of irrelevant documents in the collection is not taken into account
 - recall is undefined when there is no relevant document in the collection
 - precision is undefined when no document is retrieved

$$\text{Fallout} = \frac{\text{no. of nonrelevant items retrieved}}{\text{total no. of nonrelevant items in the collection}}$$

- Fallout can be viewed as the inverse of recall. A good system should have high recall and low fallout

4/4/01

Text-based SE

31

Total Number of Relevant Items

- In an uncontrolled environment (e.g., the web), it is unknown.
- Two possible approaches to get estimates
 - Sampling across the database and performing relevance judgment on the returned items
 - Apply different retrieval algorithms to the same database for the same query. The aggregate of relevant items is taken as the total relevant algorithm

4/4/01

Text-based SE

32

Computation of Recall and Precision

n	doc #	relevan	Recall	Precision
1	588	x	0.2	1.00
2	589	x	0.4	1.00
3	576		0.4	0.67
4	590	x	0.6	0.76
5	986		0.6	0.60
6	592	x	0.8	0.67
7	984		0.8	0.57
8	988		0.8	0.50
9	578		0.8	0.44
10	985		0.8	0.40
11	103		0.8	0.36
12	591		0.8	0.33
13	772	x	1.0	0.38
14	990		1.0	0.36

Suppose:
total no. of relevant docs = 5

$$R=1/5=0.2; \quad p=1/1=1$$

$$R=2/5=0.4; \quad p=2/2=1$$

$$R=2/5=0.4; \quad p=2/3=0.67$$

$$R=5/5=1; \quad p=5/13=0.38$$

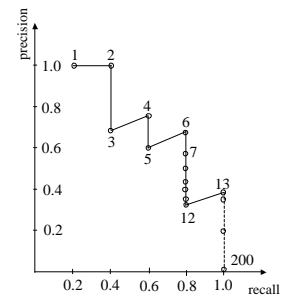
4/4/01

Text-based SE

33

Computation of Recall and Precision

n	Recall	Precision
1	0.2	1.00
2	0.4	1.00
3	0.4	0.67
4	0.6	0.76
5	0.6	0.60
6	0.8	0.67
7	0.8	0.57
8	0.8	0.50
9	0.8	0.44
10	0.8	0.40
11	0.8	0.36
12	0.8	0.33
13	1.0	0.38
14	1.0	0.36



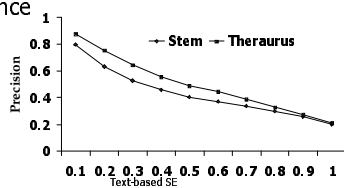
4/4/01

Text-based SE

34

Compare Two or More Systems

- Computing recall and precision values for two or more systems
- Superimposing the results in the same graph
- The curve closest to the upper right-hand corner of the graph indicates the best performance



4/4/01

Text-based SE

35

The TREC Benchmark

- TREC: Text Retrieval Conference
 - Originated from the TIPSTER program sponsored by Defense Advanced Research Projects Agency (DARPA)
 - Became an annual conference in 1992, co-sponsored by the National Institute of Standards and Technology (NIST) and DARPA
 - Participants are given parts of a standard set of documents and queries in different stages for testing and training
 - Participants submit the P/R values on the final document and query set and present their results in the conference

<http://trec.nist.gov/>

4/4/01

Text-based SE

36

The TREC Objectives

- Give a common ground for comparing different IR techniques
 - same set of documents and queries, and same evaluation method
- Sharing of resources and experiences in developing the benchmark
 - with major sponsorship from government to develop large benchmark collections
- Encourage participation from industry and academia
- Development of new evaluation techniques, particularly for new applications
 - retrieval, routing/filtering, non-English collection, web-based collection

4/4/01

Text-based SE

37

TREC Advantages

- Large scale (compared to a few Mbytes in the Cornell Collection)
- Relevance Judgements provided
- Under continuous development and with support from US Government
- Wide participation (TREC 1: 28 papers 360 pages; TREC 4: 37 papers 560 pages; TREC 7: 61 papers 600 pages; TREC 8: 74 papers)

4/4/01

Text-based SE

38

TREC Tasks

- Ad hoc - new questions are being asked on the static set of data. (library catalog searching)
- Routing - same questions are being asked, but the new information is being searched. (news clipping, library profiling)
- New tasks added after TREC 5 – Web, Interactive, multilingual, natural language, multiple database merging, filtering, very large corpus (20 Gbytes of 7.5 million documents)

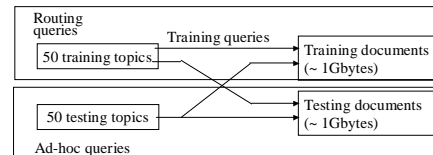
4/4/01

Text-based SE

39

TREC Collections

- TREC evaluates both Ad hoc and routing queries and provides both training and test collections:
 - 50 training topics + 1 Gbytes of training documents + relevance judgement
 - 50 training topics + 1 Gbytes of test documents
 - Adhoc: runs the 50 test topics on **both** training and test documents
 - Routing: trains the system with training queries and documents and then tests it on the unseen test **documents**.



4/4/01

Text-based SE

40

Characteristics of the TREC Collection

- both long and short documents (from a few hundred to over one thousand unique terms in a document)
- test documents consist of:

WSJ	Wall Street Journal articles (1986-1992)	550 M
AP	Associate Press Newswire (1989)	514 M
ZIFF	Computer Select Disks (Ziff-Davis Publishing)	493 M
FR	Federal Register	469 M
DOE	Abstracts from Department of Energy reports	190 M

4/4/01

Text-based SE

41

Evaluation

- Summary table statistics: number of topics, number of documents retrieved, number of relevant documents
- Recall-precision average: average precision at 11 recall levels (0 to 1 at 0.5 increment)
- Document level average: average precision when 5, 10, .., 100, ... 1000 documents are retrieved
- Average precision histogram: average precision for each topic against the medium precision of

4/4/01

Text-based SE

42

Ad hoc results — Fujitsu Laboratories, Ltd.

Summary Statistics	
Run Number:	Fub&Kuf2
Run Description:	Automatic, title + desc
Number of Topics:	50
Retrieval number of documents over all topics	
Retrieved:	8000
Relevant:	478
Ret. rate:	2990

Retrieval Level Precision Averages		Document Level Averages	
Recall	Precision	A: 5 docs	Precision
0.00	0.1796	A: 5 docs	0.5680
0.10	0.5490	A: 10 docs	0.4880
0.20	0.6177	A: 15 docs	0.4587
0.30	0.2854	A: 20 docs	0.4200
0.40	0.2997	A: 30 docs	0.3867
0.50	0.2950	A: 100 docs	0.3590
0.60	0.2291	A: 200 docs	0.1777
0.70	0.1745	A: 500 docs	0.3021
0.80	0.1381	A: 1000 docs	0.0598
0.90	0.0790		
1.00	0.0294		
Average precision over all relevant docs		R Precision (precision after R docs retrieved; where R is the number of relevant documents)	
non-interpolated: 0.2930		Exact: 0.2009	

4/4/01 Text-based SE 43

Interactive Search Engines

- Aims to improve their search results incrementally,
 - often applies to query "Find all/sites with certain property"
 - Content based Multimedia search: given a photo, find all other photos similar to it
 - Large vector space
 - Question: which feature (keyword) is important?
- Procedure:
 - User submits query
 - Engine returns result
 - User marks some returned result as relevant or irrelevant, and continues search
 - Engine returns new results
 - Iterates until user satisfied

4/4/01 Text-based SE 44

Query Reformulation

- Based on user's feedback on returned results
 - Documents that are relevant D_R
 - Documents that are irrelevant D_N
 - Build a new query vector Q' from Q
 - $\langle w_1, w_2, \dots, w_t \rangle \rightarrow \langle w_1', w_2', \dots, w_t' \rangle$
 - Best known algorithm: Rocchio's algorithm
 - Also extensively used in multimedia search

4/4/01 Text-based SE 45

Query Modification

- Using the previously identified relevant and nonrelevant document set D_R and D_N to repeatedly modify the query to reach optimality
- Starting with an initial query in the form of

$$Q' = \alpha * Q + \left(\frac{1}{R} \sum_{i \in D_R} D_i\right) - \gamma \left(\frac{1}{N} \sum_{j \in D_N} D_j\right)$$
 where Q is the original query, and α , β , and γ are suitable constants

4/4/01 Text-based SE 46

An Example

Q: original query
 D1: relevant doc.
 D2: non-relevant doc.
 $\alpha = 1, \beta = 1/2, \gamma = 1/4$
 Assume: dot-product similarity measure

	T1	T2	T3	T4	T5
Q	5	0	3	0	1
D1	2	1	2	0	0
D2	1	0	0	0	2

$$S(Q, D_i) = \sum_{j=1}^5 \langle Q_j \times D_{ij} \rangle$$

$\text{Sim}(Q, D1) = (5 \cdot 2) + (0 \cdot 1) + (3 \cdot 2) + (0 \cdot 0) + (1 \cdot 0) = 16$
 $\text{Sim}(Q, D2) = (5 \cdot 1) + (0 \cdot 0) + (3 \cdot 0) + (0 \cdot 0) + (1 \cdot 2) = 7$

4/4/01 Text-based SE 47

Example (Cont.)

$$Q' = Q + \frac{1}{2} \left(\sum_{i \in D_R} D_i \right) - \frac{1}{4} \left(\frac{1}{N} \sum_{j \in D_N} D_j \right)$$

$$Q' = (5, 0, 3, 0, 1) + \frac{1}{2} (2, 1, 2, 0, 0) - \frac{1}{4} (1, 0, 0, 0, 2)$$

$$Q' = (5.75, 0.5, 4, 0, 0.5)$$

$$Q' = (5.75, 0.5, 4, 0, 0.5)$$

New Similarity Scores:

$\text{Sim}(Q' D1) = (5.75 \cdot 2) + (0.5 \cdot 1) + (4 \cdot 2) + (0 \cdot 0) + (0.5 \cdot 0) = 20$
 $\text{Sim}(Q' D2) = (5.75 \cdot 1) + (0.5 \cdot 0) + (4 \cdot 0) + (0 \cdot 0) + (0.5 \cdot 2) = 6.75$

4/4/01 Text-based SE 48