







Web Document Modeling

Peter Brusilovsky

With slides from Jae-wook Ahn and Jampol Polvichai

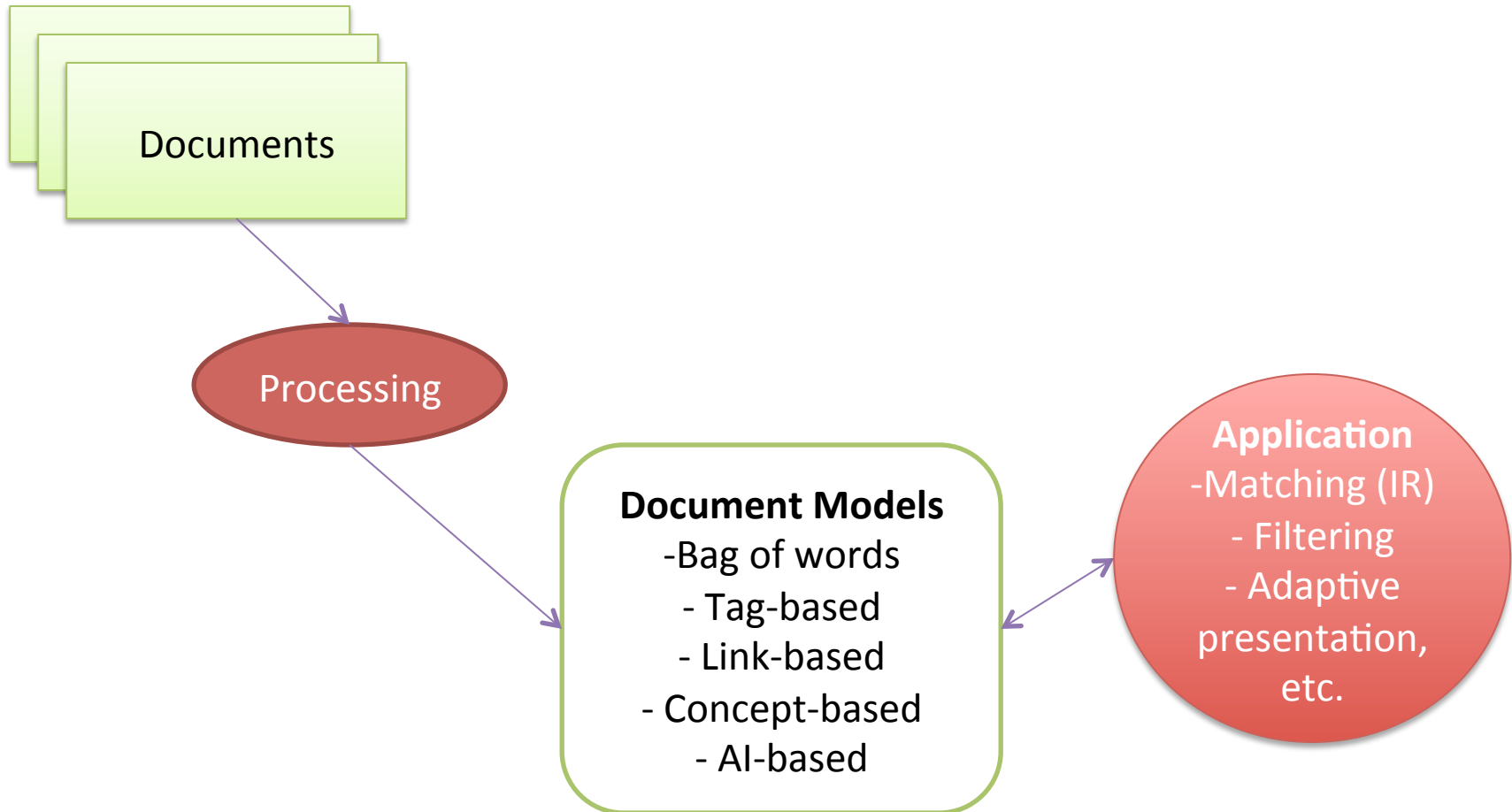
Where we are?

	Search	Navigation	Recommendation
Content-based			
Semantics / Metadata			
Social			

Introduction

- Modeling means “**the construction of an abstract representation of the document**”
 - Useful for all applications aimed at processing information automatically.
- Why build models of documents?
 - To guide the users to the right documents we need to know what they are about, what is their structure
 - Some adaptation techniques can operate with documents as “black boxes”, but others are based on the ability to understand and model documents

Document modeling



Document model

Example

the death toll rises in the middle east as the worst violence in four years spreads beyond jerusalem. the stakes are high, the race is tight. prepping for what could be a decisive moment in the presidential battle.

how a tiny town in iowa became a booming melting pot and the image that will not soon fade.

the man who captured it tells the story behind it.

	docid	term	tfidf
	ABC20001001.1830.0001	battle	3.21872
	ABC20001001.1830.0001	boom	5.01961
	ABC20001001.1830.0001	capture	4.03996
	ABC20001001.1830.0001	death	2.6322
	ABC20001001.1830.0001	decisive	4.50314
	ABC20001001.1830.0001	east	2.14989
	ABC20001001.1830.0001	fade	5.48334
	ABC20001001.1830.0001	high	1.66849
	ABC20001001.1830.0001	image	3.37131
	ABC20001001.1830.0001	iowa	5.31571
	ABC20001001.1830.0001	jerusalem	3.05699
	ABC20001001.1830.0001	man	2.68728
	ABC20001001.1830.0001	melt	5.96138
	ABC20001001.1830.0001	middle	2.51962
	ABC20001001.1830.0001	moment	3.21432
	ABC20001001.1830.0001	pot	5.50355
	ABC20001001.1830.0001	prep	7.48582
	ABC20001001.1830.0001	presidential	2.62357
	ABC20001001.1830.0001	race	3.06877
	ABC20001001.1830.0001	rise	2.95104
	ABC20001001.1830.0001	spread	3.41333

Outline

- **Classic IR based representation**
 - Preprocessing
 - Boolean, Probabilistic, Vector Space models
- **Web-IR document representation**
 - Tag based document models
 - Link based document models – HITS, Google Rank
- **Concept-based document modeling**
 - LSI
- **AI-based document representation**
 - ANN, Semantic Network, Bayesian Network

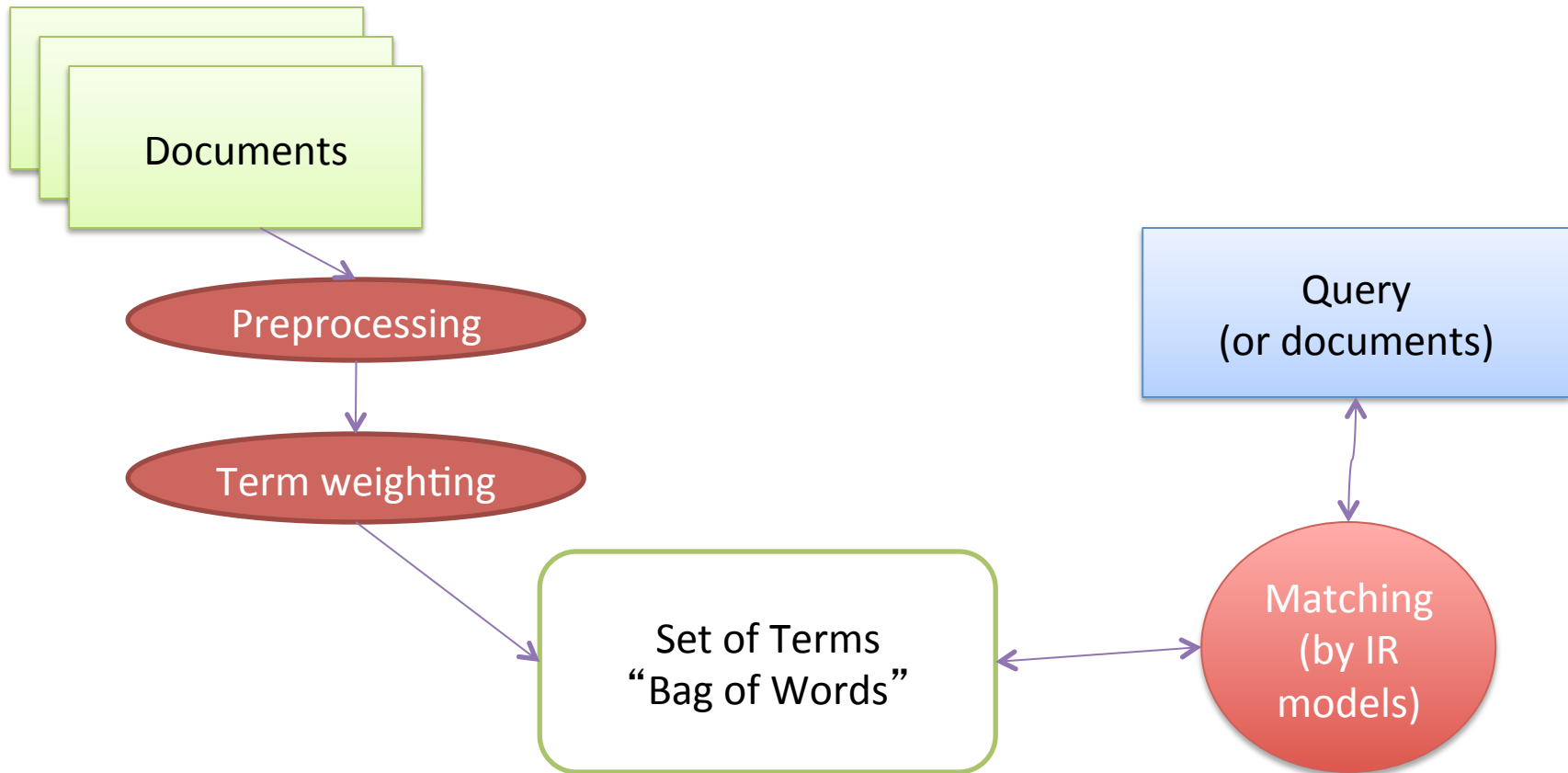
Markup Languages

A Markup Language is a text-based language that combines content with its metadata. ML support structure modeling

- **Presentational Markup**
 - Express document structure via the visual appearance of the whole text of a particular fragment.
 - Exp. Word processor
- **Procedural Markup**
 - Focuses on the presentation of text, but is usually visible to the user editing the text file, and is expected to be interpreted by software following the same procedural order in which it appears.
 - Exp. Tex, PostScript
- **Descriptive Markup**
 - Applies labels to fragments of text without necessarily mandating any particular display or other processing semantics.
 - Exp. SGML, XML

Classic IR model

Process



Preprocessing

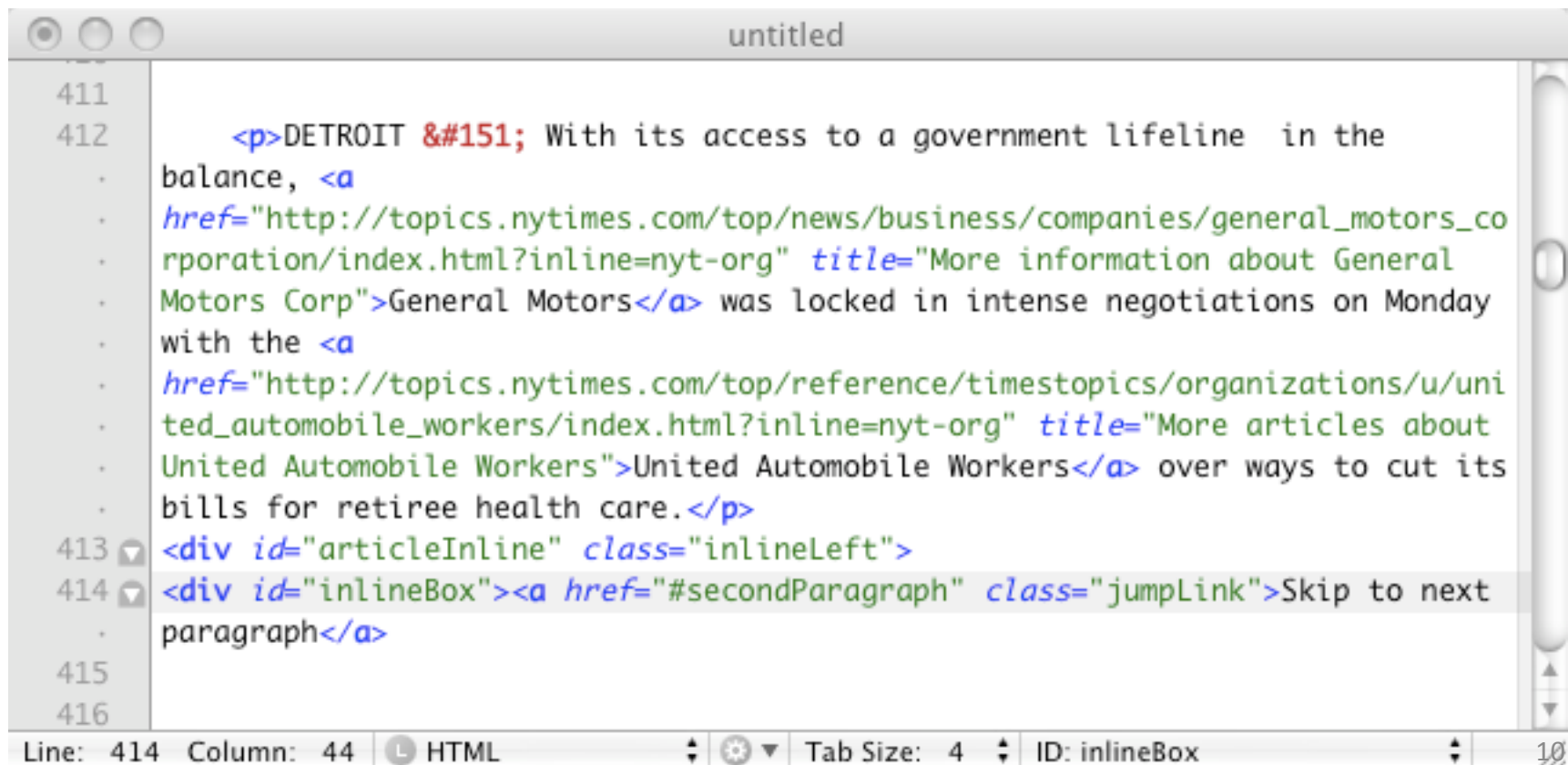
Motivation

- Extract document content itself to be processed (used)
- Remove control information
 - Tags, script, style sheet, etc
- Remove non-informative fragments
 - Stopwords, word-stems
- Possible extraction of semantic information (concepts, named entities)

Preprocessing

HTML tag removal

- Removes `<.*>` parts from the HTML document (source)



```
411
412     <p>DETROIT &#151; With its access to a government lifeline in the
    balance, <a
    href="http://topics.nytimes.com/top/news/business/companies/general_motors_co
    rporation/index.html?inline=nyt-org" title="More information about General
    Motors Corp">General Motors</a> was locked in intense negotiations on Monday
    with the <a
    href="http://topics.nytimes.com/top/reference/timestopics/organizations/u/uni
    ted_automobile_workers/index.html?inline=nyt-org" title="More articles about
    United Automobile Workers">United Automobile Workers</a> over ways to cut its
    bills for retiree health care.</p>
413 <div id="articleInline" class="inlineLeft">
414 <div id="inlineBox"><a href="#secondParagraph" class="jumpLink">Skip to next
    paragraph</a>
415
416
```

Line: 414 Column: 44 HTML Tab Size: 4 ID: inlineBox 10

Preprocessing

HTML tag removal

DETROIT — With its access to a government lifeline in the balance, General Motors was locked in intense negotiations on Monday with the United Automobile Workers over ways to cut its bills for retiree health care.

Preprocessing

Tokenizing/case normalization

- Extract term/feature tokens from the text

detroit with its access to a government lifeline in the balance general motors was locked in intense negotiations on monday with the united automobile workers over ways to cut its bills for retiree health care

Preprocessing

Stopword removal

- Very common words
- Do not contribute to separate a document from another meaningfully
- Usually a standard set of words are matched/removed

detroit ~~with its~~ access ~~to a~~ government
lifeline ~~in the~~ balance general motors ~~was~~
locked ~~in~~ intense negotiations ~~on~~
monday ~~with the~~ united automobile
workers ~~over~~ ways to ~~cut its~~ bills ~~for~~
retiree health care

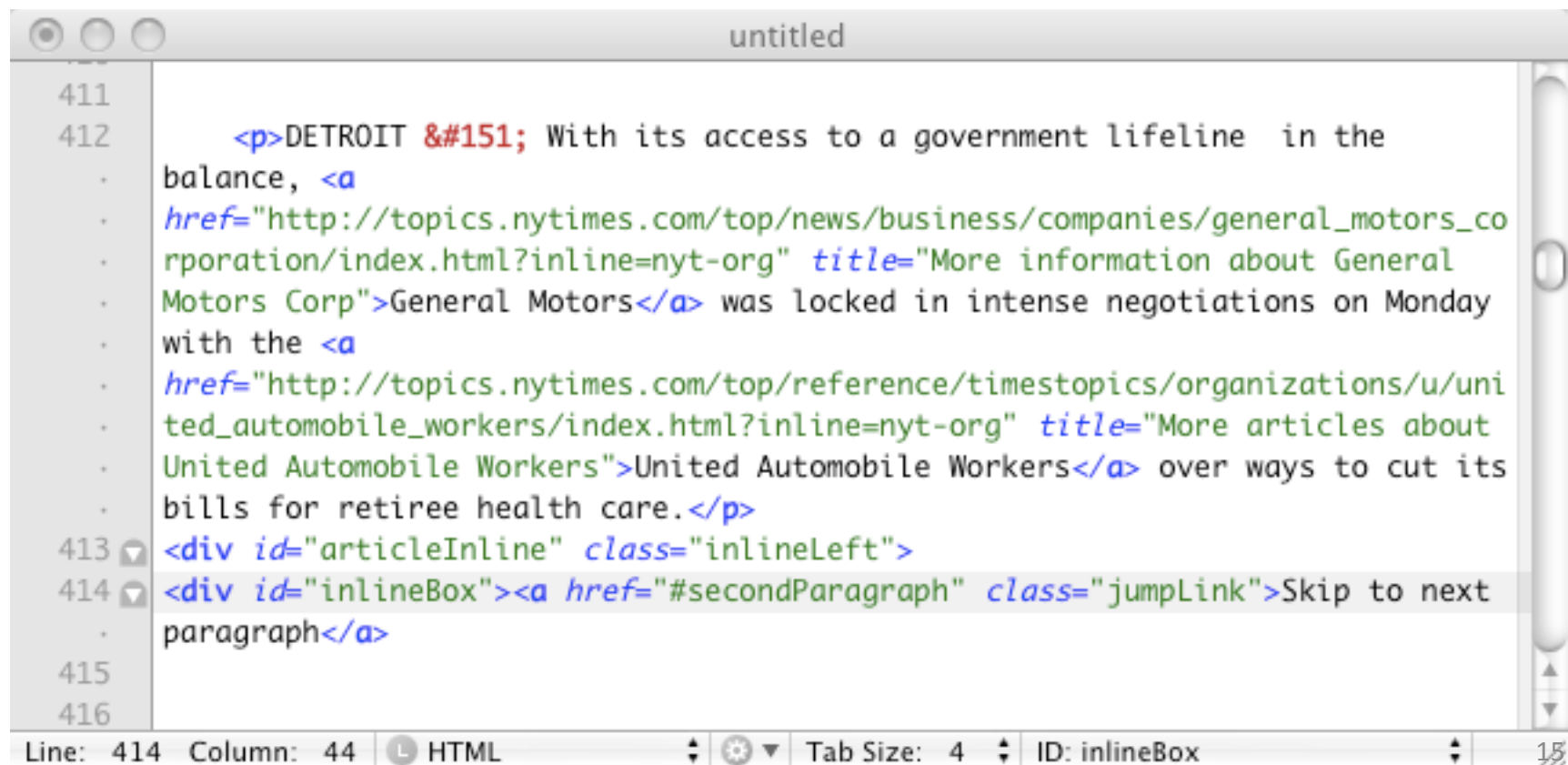
Named Entities

- Named Entities and other concepts are typically extracted from full text

DETROIT — With its access to a government lifeline in the balance, **General Motors** was locked in intense negotiations on **Monday** with the **United Automobile Workers** over ways to cut its bills for retiree health care.

Extracting Semantic Information

- Some times HTML tags are useful



```
411
412     <p>DETROIT &#151; With its access to a government lifeline in the
    balance, <a
    href="http://topics.nytimes.com/top/news/business/companies/general_motors_co
    rporation/index.html?inline=nyt-org" title="More information about General
    Motors Corp">General Motors</a> was locked in intense negotiations on Monday
    with the <a
    href="http://topics.nytimes.com/top/reference/timestopics/organizations/u/uni
    ted_automobile_workers/index.html?inline=nyt-org" title="More articles about
    United Automobile Workers">United Automobile Workers</a> over ways to cut its
    bills for retiree health care.</p>
413 <div id="articleInline" class="inlineLeft">
414 <div id="inlineBox"><a href="#secondParagraph" class="jumpLink">Skip to next
    paragraph</a>
415
416
```

Line: 414 Column: 44 HTML Tab Size: 4 ID: inlineBox 15

Preprocessing

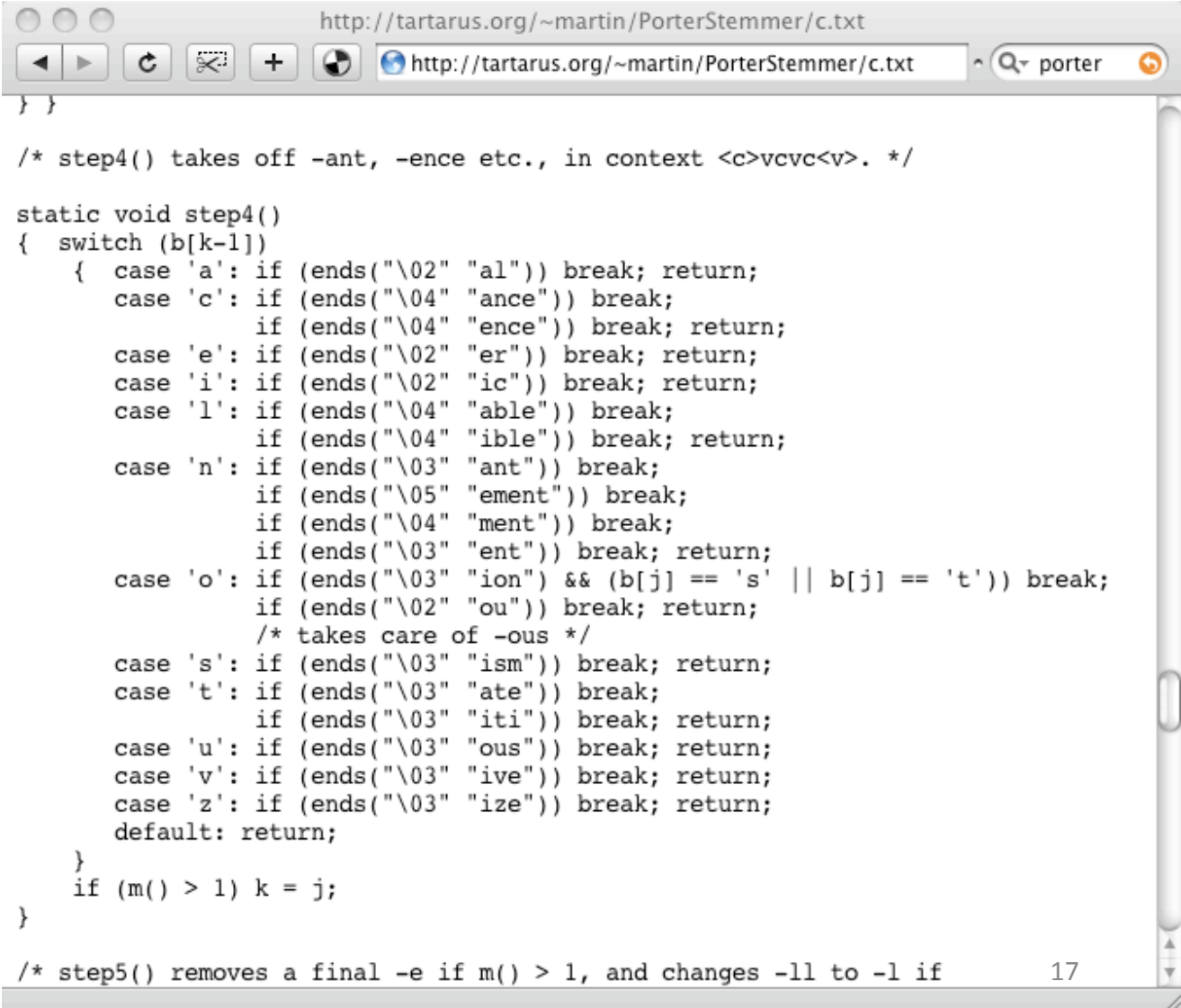
Stemming

- Extracts word “stems” only
- Avoid word variation that are not informative
 - apples, apple
 - Retrieval, retrieve, retrieving
 - *Should they be distinguished? Maybe not.*
- Porter
- Krovetz

Preprocessing

Stemming (Porter)

- Martin Porter, 1979
- Cyclical recognition and removal of known suffixes and prefixes
- Try Demo at http://qaa.ath.cx/porter_js_demo.html



The screenshot shows a web browser window with the address bar displaying `http://tartarus.org/~martin/PorterStemmer/c.txt`. The browser's search bar contains the word "porter". The main content area displays the C code for the Porter Stemmer algorithm. The code includes a comment for `step4()` and a function definition for `step4()` that uses a switch statement to handle various suffixes. At the bottom, a comment for `step5()` is visible.

```
}}  
  
/* step4() takes off -ant, -ence etc., in context <c>vcvc<v>. */  
  
static void step4()  
{  
    switch (b[k-1])  
    {  
        case 'a': if (ends("\02" "al")) break; return;  
        case 'c': if (ends("\04" "ance")) break;  
                   if (ends("\04" "ence")) break; return;  
        case 'e': if (ends("\02" "er")) break; return;  
        case 'i': if (ends("\02" "ic")) break; return;  
        case 'l': if (ends("\04" "able")) break;  
                   if (ends("\04" "ible")) break; return;  
        case 'n': if (ends("\03" "ant")) break;  
                   if (ends("\05" "ement")) break;  
                   if (ends("\04" "ment")) break;  
                   if (ends("\03" "ent")) break; return;  
        case 'o': if (ends("\03" "ion") && (b[j] == 's' || b[j] == 't')) break;  
                   if (ends("\02" "ou")) break; return;  
                   /* takes care of -ous */  
        case 's': if (ends("\03" "ism")) break; return;  
        case 't': if (ends("\03" "ate")) break;  
                   if (ends("\03" "iti")) break; return;  
        case 'u': if (ends("\03" "ous")) break; return;  
        case 'v': if (ends("\03" "ive")) break; return;  
        case 'z': if (ends("\03" "ize")) break; return;  
        default: return;  
    }  
    if (m() > 1) k = j;  
}  
  
/* step5() removes a final -e if m() > 1, and changes -ll to -l if
```

Preprocessing

Stemming (Krovetz)

- Bob Krovetz, 1993
- Makes use of inflectional linguistic morphology
- Removes inflectional suffixes in three steps
 - single form (e.g. ‘-ies’ , ‘-es’ , ‘-s’)
 - past to present tense (e.g. ‘-ed’)
 - removal of ‘-ing’
- Checking in a dictionary
- More human-readable

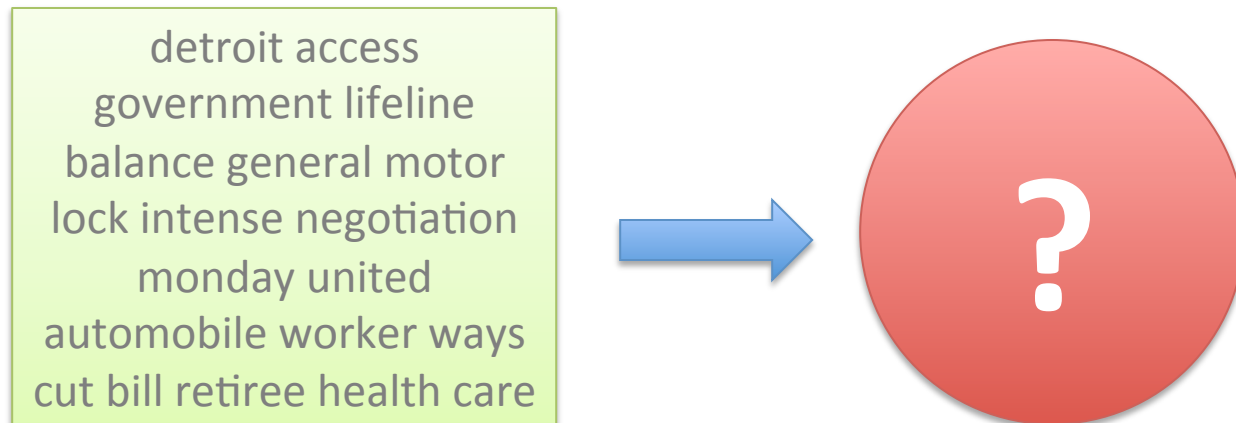
Preprocessing

Stemming

- Porter stemming example
[detroit access govern lifelin balanc gener motor
lock intens negoti mondai unit automobil worker
wai cut bill retire health care]
- Krovetz stemming example
[detroit access government lifeline balance
general motor lock intense negotiation monday
united automobile worker ways cut bill retiree
health care]

Term weighting

- How should we represent the terms/features after the processes so far?



Term weighting

Document-term matrix

- Columns – every term appeared in the corpus (not a single document)
- Rows – every document in the collection
- Example
 - If a collection has N documents and M terms...

	T1	T2	T3	...	T _M
Doc 1	0	1	1	...	0
Doc 2	1	0	0	...	1
...
Doc N	0	0	0	...	1

Term weighting

Document-term matrix

- Document-term matrix
 - Binary (if appears 1, otherwise 0)

	attract	benefit	book	...	zoo
Doc 1	0	1	1	...	0
Doc 2	1	0	0	...	1
Doc 3	0	0	0	...	1

- Every term is treated equivalently

Term weighting

Term frequency

- So, we need “***weighting***”
 - Give different “***importance***” to different terms
- TF
 - Term frequency
 - How many times a term appeared in a document?
 - Higher frequency → higher relatedness

Term weighting

Term frequency

untitled 3	
1	document 342
2	term 182
3	1 170
4	model 164
5	web 151
6	5 136
7	0 128
8	represent 102
9	weight 85
10	system 78
11	page 69
12	ti 68
13	base 65
14	retrieval 63
15	technique 61
16	network 61
17	query 57
18	information 57
19	html 54
20	ir 53
21	collect 48
22	set 45
23	3 44
24	vector 43
25	illustrate 43
Line: 25 Column: 14 Plain Text	

untitled 3	
26	springe 41
27	use 40
28	text 39
29	process 39
30	index 38
31	compute 38
32	2 38
33	relevant 36
34	method 36
35	dj 36
36	chapter 36
37	tag 35
38	link 35
39	rank 34
40	calculate 34
41	node 33
42	space 30
43	semantic 30
44	fig 30
45	subsection 29
46	concept 29
47	algorithm 29
48	trec 28
49	read 28
50	matrix 28
51	conference 28
Line: 51 Column: 14 Plain Text	

Term weighting

IDF

- IDF
 - Inverse Document Frequency
 - Generality of a term → too general, not beneficial
 - Example
 - **“Information”** (in ACM Digital Library)
 - 99.99% of articles will have it
 - TF will be very high in each document, IDF low
 - **“Personalization”**
 - Say, 1% of documents will have it
 - TF again will be very high, IDF high

Term weighting

IDF

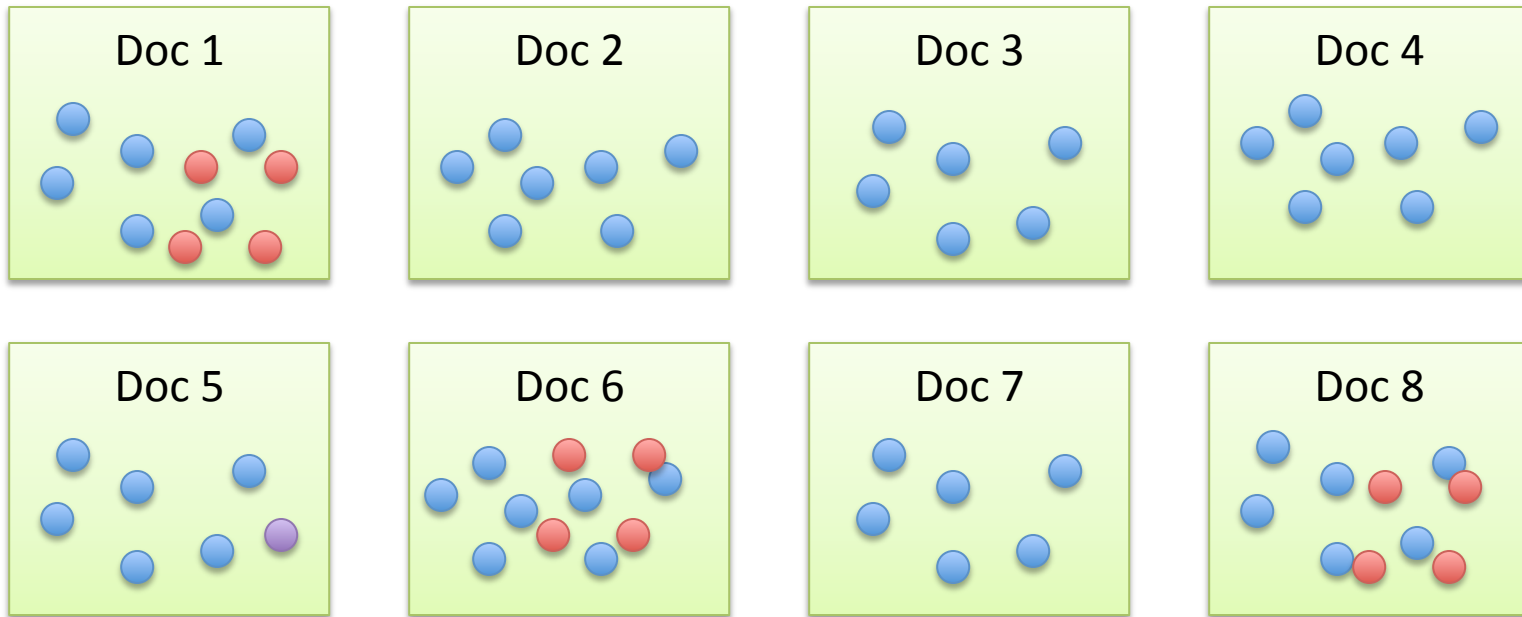
- IDF

$$\log\left(\frac{N}{DF}\right)$$

- N = number of documents in the corpus
- DF = document frequency = number of documents that have the term
- If ACM DL has 1M documents
 - $IDF(\text{“information”}) = \log(1M/999900) = 0.0001$
 - $IDF(\text{“personalization”}) = \log(1M/50000) = 30.63$

Term weighting

IDF



- information, personalization, recommendation
- Can we say...
 - Doc 1, 2, 3... are about information?
 - Doc 1, 6, 8... are about personalization?
 - Doc 5 is about recommendation?

Term weighting

TF*IDF

- TF*IDF
 - TF multiplied by IDF
 - Considers TF and IDF at the same time
 - High frequency terms focused in smaller portion of documents gets higher scores

Document	benef	attract	sav	springer	book
d1	0.176	0.176	0.417	0.176	0.176
d2	0.000	0.350	0.000	0.528	0.000
d3	0.528	0.000	0.000	0.000	0.176

Term weighting

BM25

- OKAPI BM25
 - Okapi Best Match 25
 - Probabilistic model – calculates term relevance within a document
 - Computes a term weight according to the probability of its appearance in a relevant document and to the probability of it appearing in a non-relevant document in a collection D

$$w_i = TF(t_i, d) \frac{\log \frac{(|D| - TF(t_i, d) + 0,5)}{(|D| + 0,5)}}{k_1 \cdot ((1 - b) + b \cdot \frac{|d|}{d}) + TF(t_i, d)}$$

Term weighting

Entropy weighting

- Entropy weighting

$$w_i = \log(TF(t_i, d) + 1) \left(1 + \frac{1}{\log(|D|)} \sum_{j=1}^{|D|} \left[\frac{TF(t_i, d_j)}{DF(t_i)} \log \frac{TF(t_i, d_j)}{DF(t_i)} \right] \right)$$

- Entropy of term t_i
 - -1 : equal distribution of all documents
 - 0 : appearing only 1 document

IR models

- Boolean
- Probabilistic
- Vector Space

IR Models

Boolean model

ITD	Terms	d_1	d_2	d_3
t_1	benefit	1	0	0
t_2	attract	1	1	0
t_3	save	1	1	0
t_4	springer	1	1	1
t_5	book	1	0	1
t_6	sign	0	0	1
t_7	email	1	1	0
t_8	titl	0	1	1
t_9	info	0	0	1
t_{10}	special	0	0	1
t_{11}	announc	0	0	1
t_{12}	alert	0	0	1
t_{13}	pai	0	1	0
t_{14}	inform	1	0	1
t_{15}	notif	0	1	1
t_{16}	servic	1	0	1

- Based on set theory and Boolean algebra

$$q = \text{springer} \wedge (\text{inform} \vee \text{info})$$

- $\rightarrow d_1, d_3$

IR Models

Boolean model

- Simple and easy to implement
- Shortcomings
 - Only retrieves exact matches
 - No partial match
 - No ranking
 - Depends on user query formulation

IR Models

Probabilistic model

- Binary weight vector
- Query-document similarity function
- Probability that a certain document is relevant to a certain query
- Ranking – according to the probability to be relevant

IR Models

Probabilistic model

- Similarity calculation

$$\text{sim}(d_j, q) = \frac{P(R/d_j)}{P(\bar{R}/d_j)}$$

$$\text{sim}(d_j, q) \sim \frac{P(d_j/R)}{P(d_j/\bar{R})} \quad \text{Bayes Theorem and removing some constants}$$

$$\text{sim}(d_j, q) \sim \sum_{i=1}^n \log \frac{P(t_i/R)P(\bar{t}_i/\bar{R})}{P(t_i/\bar{R})P(\bar{t}_i/R)}$$

$$P(d_j/R) = \prod_{t=1}^n P(t_i/R)$$

- Simplifying assumptions
 - No relevance information at startup

$$P(t_i/R) = 0.5$$

$$P(t_i/\bar{R}) = n_i/N$$

IR Models

Probabilistic model

- Shortcomings
 - Division of the set of documents into relevant / non-relevant documents
 - Term independence assumption
 - Index terms – binary weights

IR Models

Vector space model

- Document = m dimensional space (m = index terms)
- Each term represents a dimension
- Component of a document vector along a given direction \rightarrow term importance
- Query and documents are represented as vectors

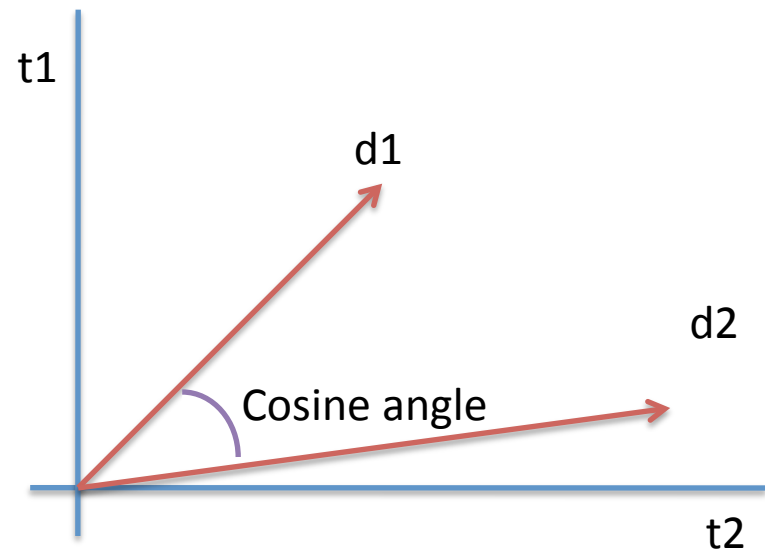
IR Models

Vector space model

- Document similarity
 - Cosine angle

$$\text{sim}(d_j, q) = \cos(\vec{d_j}, \vec{q}) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| |\vec{q}|}$$

- Benefits
 - Term weighting
 - Partial matching
- Shortcomings
 - Term independency



IR Models

Vector space model

- Example
 - Query = “springer book”
 - $q = (0, 0, 0, 1, 1)$
 - $\text{Sim}(d1, q) = (0.176 + 0.176) / (\sqrt{1} + \sqrt{0.176^2 + 0.176^2 + 0.417^2 + 0.176^2 + 0.176^2}) = 0.228$
 - $\text{Sim}(d2, q) = (0.528) / (\sqrt{1} + \sqrt{0.350^2 + 0.528^2}) = 0.323$
 - $\text{Sim}(d3, q) = (0.176) / (\sqrt{1} + \sqrt{0.528^2 + 0.176^2}) = 0.113$

Document	benef	attract	sav	springer	book
d1	0.176	0.176	0.417	0.176	0.176
d2	0.000	0.350	0.000	0.528	0.000
d3	0.528	0.000	0.000	0.000	0.176

IR Models

Vector space model

- Document – document similarity
- $\text{Sim}(d1, d2) = 0.447$
- $\text{Sim}(d2, d3) = 0.0$
- $\text{Sim}(d1, d3) = 0.408$

Document	benef	attract	sav	springer	book
d1	0.176	0.176	0.417	0.176	0.176
d2	0.000	0.350	0.000	0.528	0.000
d3	0.528	0.000	0.000	0.000	0.176

Curse of dimensionality

- TDT4
 - $|D| = 96,260$
 - $|ITD| = 118,205$
- If linearly calculates $\text{sim}(q, D)$
 - $96,260$ (per each document) * $118,205$ (inner product) comparisons
- However, document matrices are very sparse
 - Mostly 0's
 - Space, calculation inefficient to store those 0's

Curse of dimensionality

- Inverted index
 - Index from term to document

Query			
docid	term	tfidf	
ABC20001001.1830.0001	battle	3.21872	
ABC20001001.1830.0001	boom	5.01961	
ABC20001001.1830.0001	capture	4.03996	
ABC20001001.1830.0001	death	2.6322	
ABC20001001.1830.0001	decisive	4.50314	
ABC20001001.1830.0001	east	2.14989	
ABC20001001.1830.0001	fade	5.48334	
ABC20001001.1830.0001	high	1.66849	
ABC20001001.1830.0001	image	3.37131	
ABC20001001.1830.0001	iowa	5.31571	
ABC20001001.1830.0001	jerusalem	3.05699	
ABC20001001.1830.0001	man	2.68728	
ABC20001001.1830.0001	melt	5.96138	
ABC20001001.1830.0001	middle	2.51962	
ABC20001001.1830.0001	moment	3.21432	
ABC20001001.1830.0001	pot	5.50355	
ABC20001001.1830.0001	prep	7.48582	
ABC20001001.1830.0001	presidential	2.62357	
ABC20001001.1830.0001	race	3.06877	
ABC20001001.1830.0001	rise	2.95104	
ABC20001001.1830.0001	spread	3.41332	

Web-IR document representation

- Enhances the classic VSM
- Possibilities offered by HTML languages
- Tag-based
- Link-based
 - HITS
 - PageRank

Web-IR

Tag-based approaches

- Give different weights to different tags
 - Some text fragments within a tag may be more important than others
 - <body>, <title>, <h1>, <h2>, <h3>, <a> ...

Web-IR

Tag-based approaches

- WEBOR system
- Six classes of tags

Table 5.10. The tag class hierarchy used by the WEBOR system

Class Name	HTML Tags
Anchor	A
H1-H2	H1-H2
H3-H6	H3, H4, H5, H6
Strong	STRONG, B, EM, I, U, DL, OL, UL
Title	TITLE
Plain Text	None of the above

$$w_t = (\overrightarrow{CIV} \bullet \overrightarrow{TFV})idf$$

- CIV = class importance vector
- TFV = class frequency vector

Web-IR

Tag-based approaches

- Term weighting example
 - $CIV = \{0.6, 1.0, 0.8, 0.5, 0.7, 0.8, 0.5\}$
 - $TFV(\text{"personalization"}) = \{0, 3, 3, 0, 0, 8, 10\}$
 - $W(\text{"personalization"}) = (0.0 + 3.0 + 2.4 + 0.0 + 0.0 + 6.4 + 5.0) * IDF$

Table 5.10. The tag class hierarchy used by the WEBOR system

Class Name	HTML Tags
Anchor	A
H1-H2	H1-H2
H3-H6	H3, H4, H5, H6
Strong	STRONG, B, EM, I, U, DL, OL, UL
Title	TITLE
Plain Text	None of the above

Web-IR

HITS (Hyperlink-Induced Topic Search)

- Link-based approach
- Promote search performance by considering Web document links
- Works on an initial set of retrieved documents
- Hub and authorities
 - A good authority page is one that is pointed to by many good hub pages
 - A good hub page is one that is pointed to by many good authority pages
 - ***Circular definition*** → iterative computation

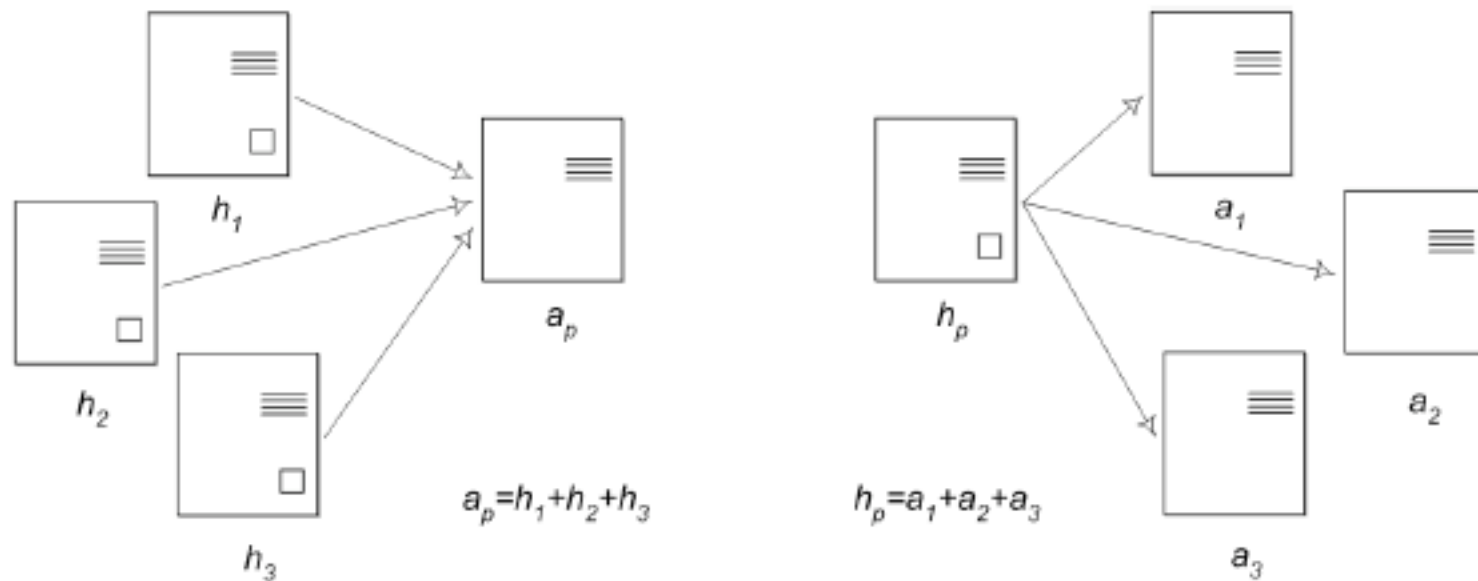
Web-IR

HITS (Hyperlink-Induced Topic Search)

- Iterative update of authority & hub vectors

$$a_p \leftarrow \sum_{q:(q,p) \in E} h_q$$

$$h_p \leftarrow \sum_{q:(p,q) \in E} a_q$$



Web-IR

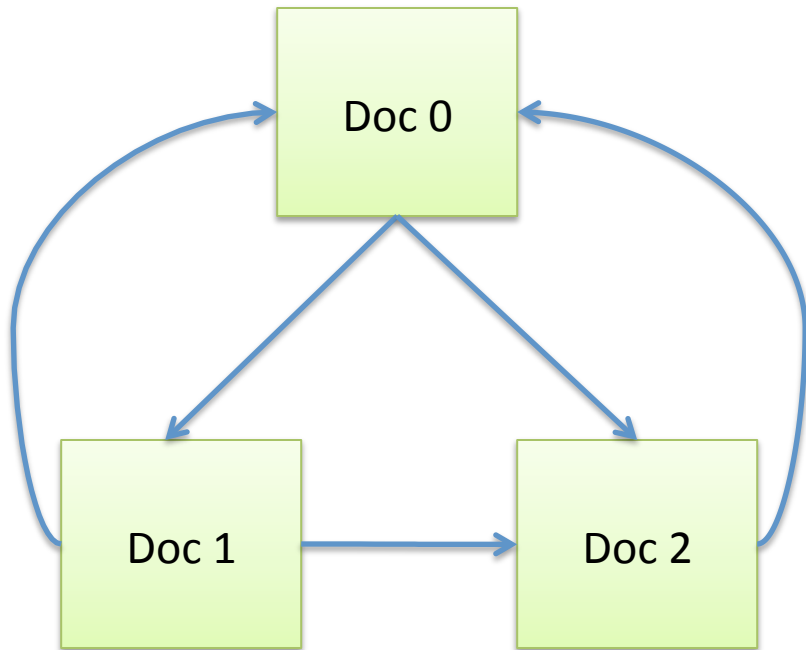
HITS (Hyperlink-Induced Topic Search)

Algorithm 1. Pseudo-code of the HITS algorithm, where authorities a_p and hubs h_p of the pages are stored in the vectors a and h .

```
 $V \leftarrow$  collection of  $n$  pages  
 $N \leftarrow$  number of iterations  
 $z \leftarrow (1, 1, \dots, 1) \in \mathbb{R}^{|V|}$   
 $a_0 \leftarrow z$   
 $h_0 \leftarrow z$   
for  $i = 0$  to  $N$  do  
  {apply Eq. 5.27 to  $(a_{i-1}; h_{i-1})$  and draw the new authority vector  $\hat{a}_i$ }  
  {apply Eq. 5.28 to  $(\hat{a}_i; h_{i-1})$  and draw the new hub vector  $\hat{h}_i$ }  
  {normalize both  $\hat{a}_i$  and  $\hat{h}_i$  to 1}  
   $a_i \leftarrow \hat{a}_i$   
   $h_i \leftarrow \hat{h}_i$   
end for
```

Web-IR

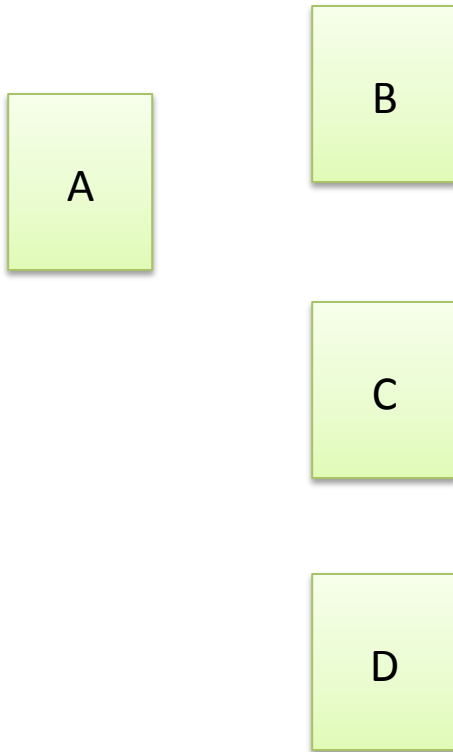
HITS (Hyperlink-Induced Topic Search)



- $N=1$
 - $A = [0.371 \ 0.557 \ 0.743]$
 - $H = [0.667 \ 0.667 \ 0.333]$
- $N = 10$
 - $A = [0.344 \ 0.573 \ 0.744]$
 - $H = [0.722 \ 0.619 \ 0.309]$
- $N = 1000$
 - $A = [0.328 \ 0.591 \ 0.737]$
 - $H = [0.737 \ 0.591 \ 0.328]$

Web-IR

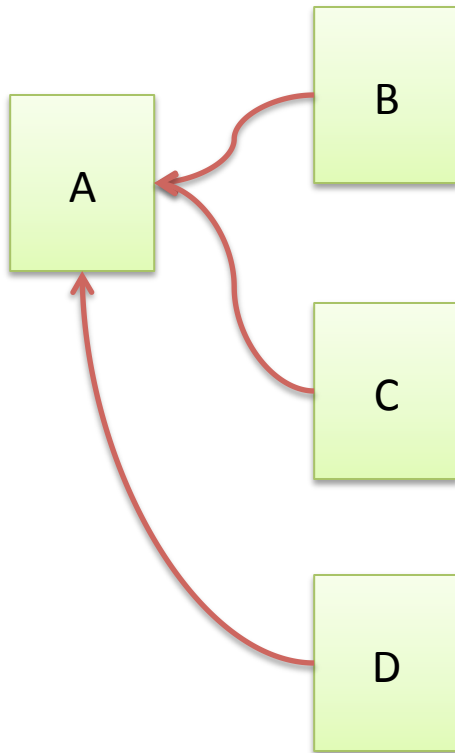
PageRank



- Google
- Unlike HITS
 - Not limited to a specific initial retrieved set of documents
 - Single value
- Initial state = no link
- Evenly divide scores to 4 documents
- $PR(A) = PR(B) = PR(C) = PR(D) = 1 / 4 = 0.25$

Web-IR

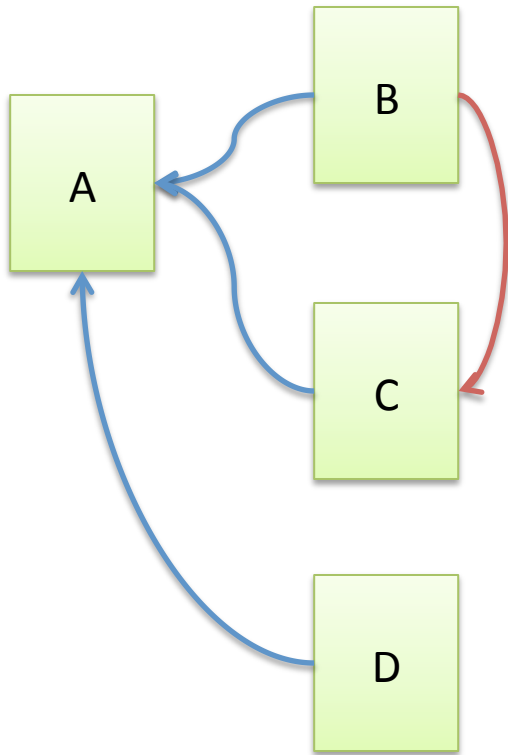
PageRank



- $PR(A)$
 $= PR(B) + PR(C) + PR(D)$
 $= 0.25 + 0.25 + 0.25$
 $= 0.75$

Web-IR

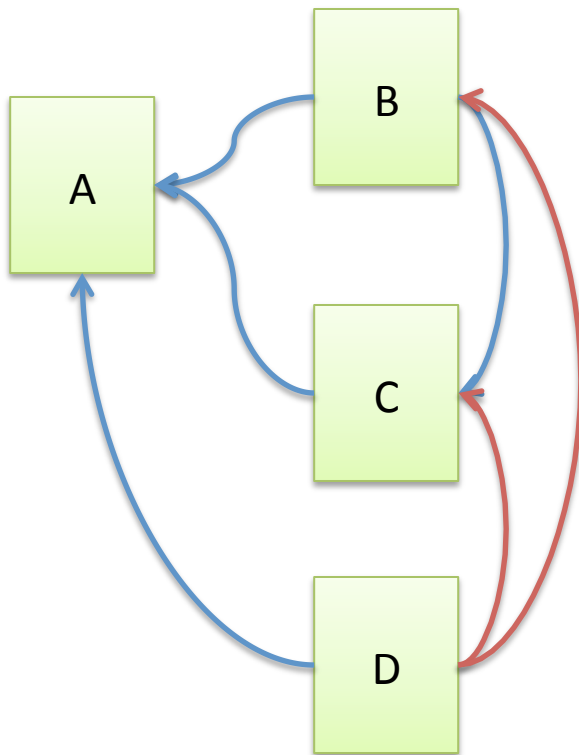
PageRank



- $PR(B) \text{ to } A = 0.25/2 = 0.125$
- $PR(B) \text{ to } C = 0.25/2 = 0.125$
- $PR(A)$
 $= PR(B) + PR(C) + PR(D)$
 $= 0.125 + 0.25 + 0.25 = 0.625$

Web-IR

PageRank



- $PR(D) \text{ to } A = 0.25/3 = 0.083$
- $PR(D) \text{ to } B = 0.25/3 = 0.083$
- $PR(D) \text{ to } C = 0.25/3 = 0.083$
- $PR(A) = PR(B) + PR(C) + PR(D) = 0.125 + 0.25 + 0.083 = 0.458$
- Recursively keep calculating to further documents linking to A, B, C, and D

Concept-based document modeling

LSI (Latent Semantic Indexing)

- Represents documents by **concepts**
 - Not by **terms**
- Reduce **term** space → **concept** space
 - Linear algebra technique : SVD (Singular Value Decomposition)
- Step (1) : Matrix decomposition – original document matrix A is factored into three matrices

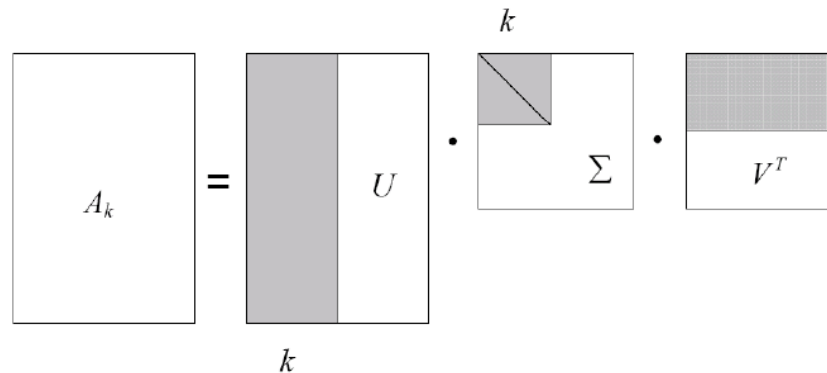
$$A = U \cdot \Sigma \cdot V^T$$

Concept-based document modeling

LSI

- Step (2) : A rank k is selected from the original equation (k = reduced # of concept space)

$$A_k = U_k \cdot \Sigma_k \cdot V_k^T$$



- Step (3) : The original term-document matrix A is converted to A_k

Concept-based document modeling

LSI

- Document-term matrix A

ITD	d_1	d_2	d_3
new	1	1	0
benefit	1	1	0
attractive	0	1	0
service	1	0	0
springer	0	0	1
info	0	1	1
special	0	0	1

Concept-based document modeling

LSI

- Decomposition

ITD	d_1	d_2	d_3
new	1	1	0
benefit	1	1	0
attractive	0	1	0
service	1	0	0
springer	0	0	1
info	0	1	1
special	0	0	1

$$A = U \cdot \Sigma \cdot V^T$$

```

-0.5566825  2.581989e-01  0.0662596
-0.5566825  2.581989e-01  0.0662596
-0.3243097 -1.110223e-16 -0.5686780
-0.2323728  2.581989e-01  0.6349376
-0.1161864 -5.163978e-01  0.3174688
-0.4404961 -5.163978e-01 -0.2512092
-0.1161864 -5.163978e-01  0.3174688

2.406509  0.000000  0.000000
0.000000  1.732051  0.000000
0.000000  0.000000  1.099414

-0.5592073 -7.804543e-01 -0.2796037
0.4472136 -1.387779e-16 -0.8944272
0.6980596 -6.252128e-01  0.3490298
    
```

Concept-based document modeling

LSI

- Low rank approximation ($k = 2$)

ITD	d_1	d_2	d_3
new	1	1	0
benefit	1	1	0
attractive	0	1	0
service	1	0	0
springer	0	0	1
info	0	1	1
special	0	0	1

$$A = U \cdot \Sigma \cdot V^T$$

-0.5566825	2.581989e-01	0.0662596
-0.5566825	2.581989e-01	0.0662596
-0.3243097	-1.110223e-16	-0.5686780
-0.2323728	2.581989e-01	0.6349376
-0.1161864	-5.163978e-01	0.3174688
-0.4404961	-5.163978e-01	-0.2512092
-0.1161864	-5.163978e-01	0.3174688
2.406509	0.000000	0.000000
0.000000	1.732051	0.000000
0.000000	0.000000	1.099414
-0.5592073	-7.804543e-01	-0.2796037
0.4472136	-1.387779e-16	-0.8944272
0.6980596	-6.252128e-01	0.3490298

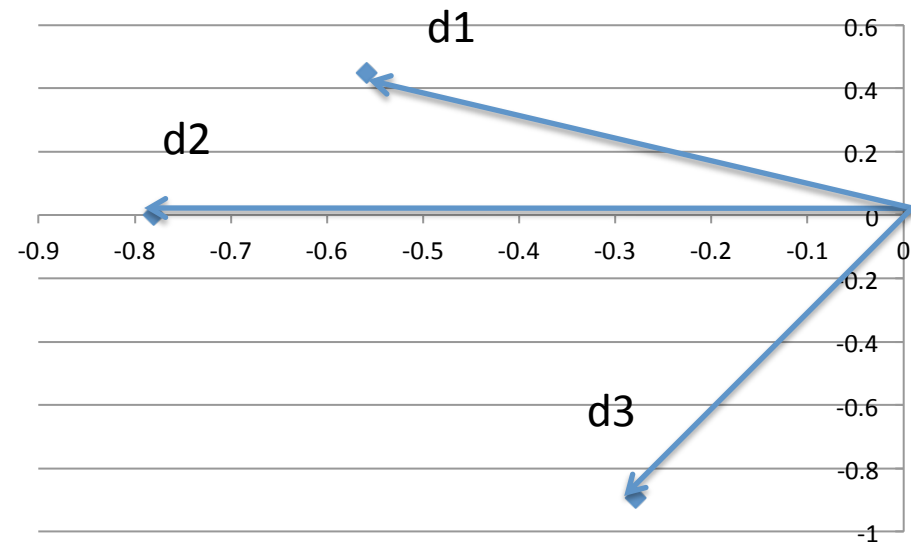
Concept-based document modeling

LSI

- Final A_k
 - Columns : documents
 - Rows : concepts (k=2)

Table 5.12. The resulting Matrix A_k

-1.3457	1.0762	1.6799
-1.3518	-0.0000	-1.0829



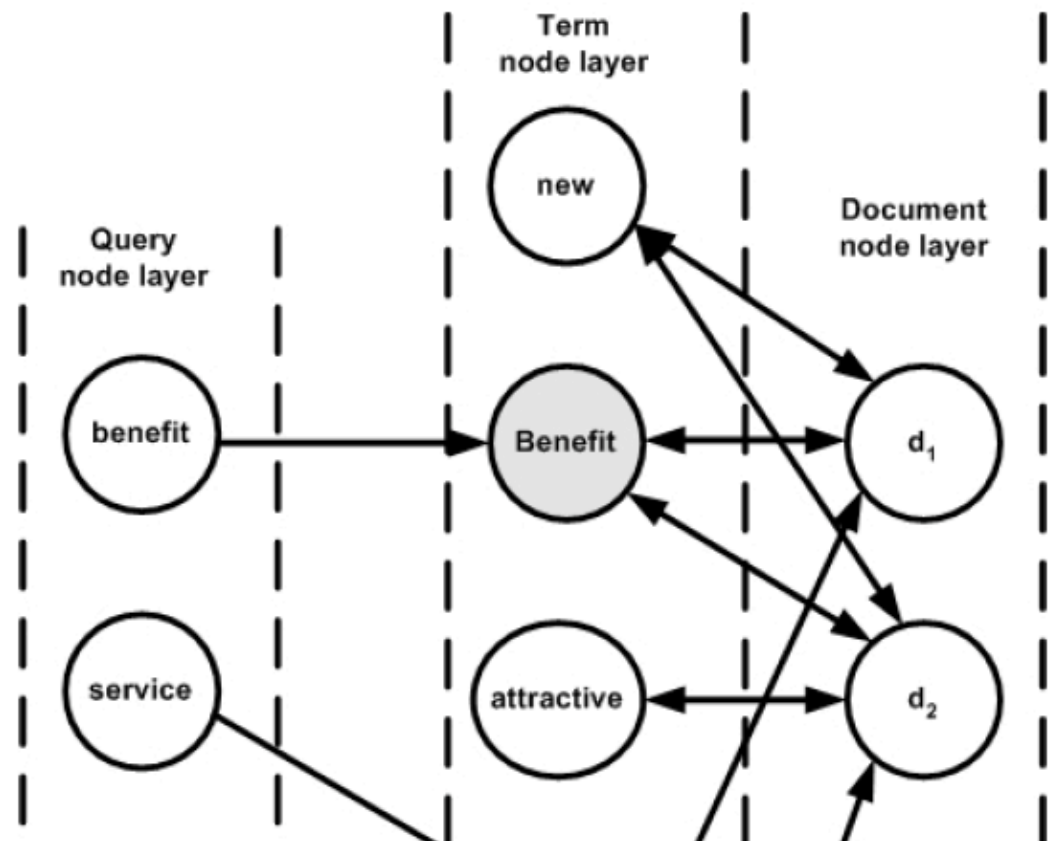
V^T = SVD Document Matrix

-0.5592073	-7.804543e-01	-0.2796037
0.4472136	-1.387779e-16	-0.8944272
0.6980596	-6.252128e-01	0.3490298

AI-based approaches

Artificial Neural Networks

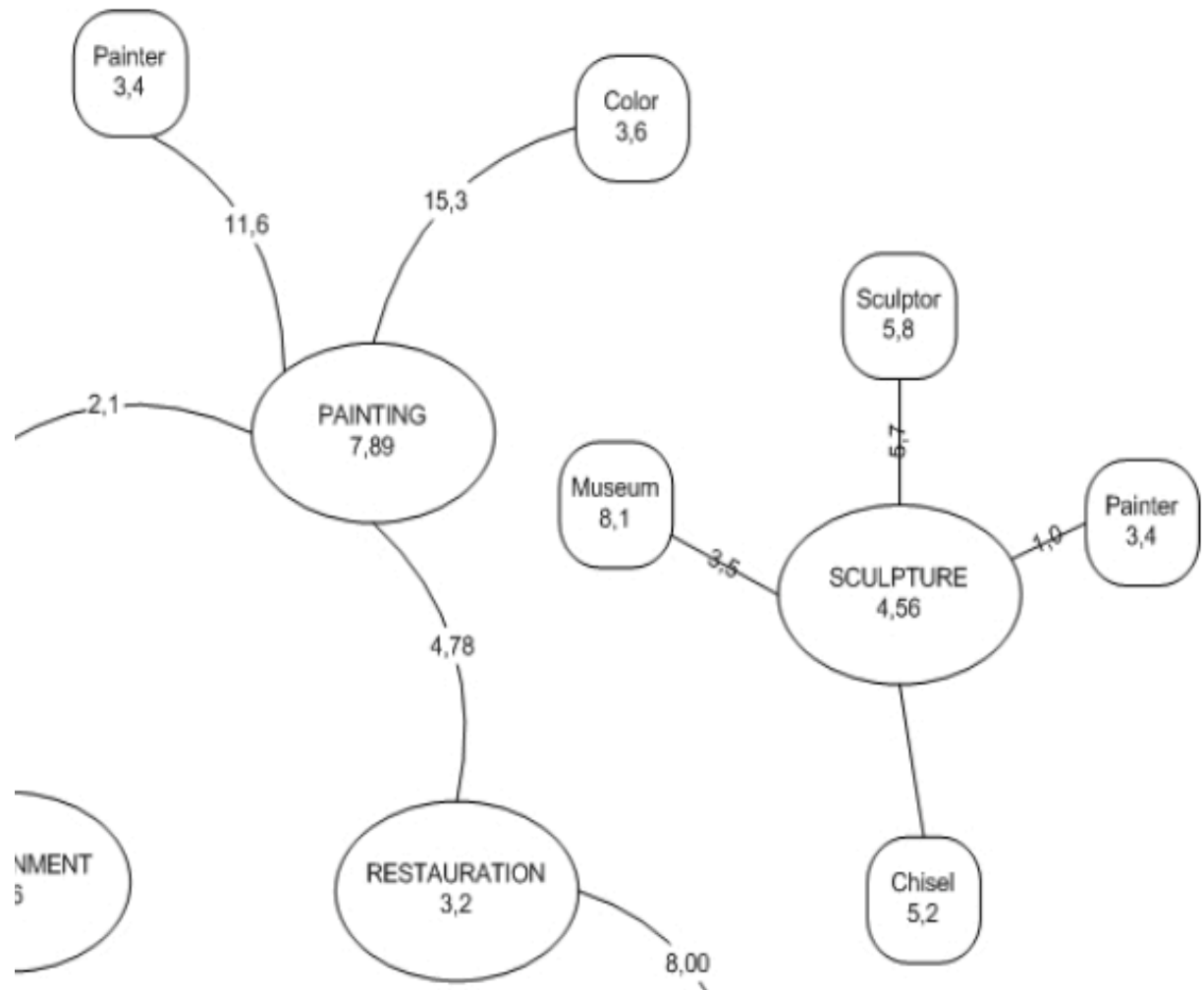
- Query, term, documents → separated into 3 layers
- Term-document weight = norm. TF-IDF
- Query → term activation → sum of the signals exceeds a threshold → document retrieval



AI-based approaches

Semantic Networks

- Conceptual knowledge
- Relationship between concepts



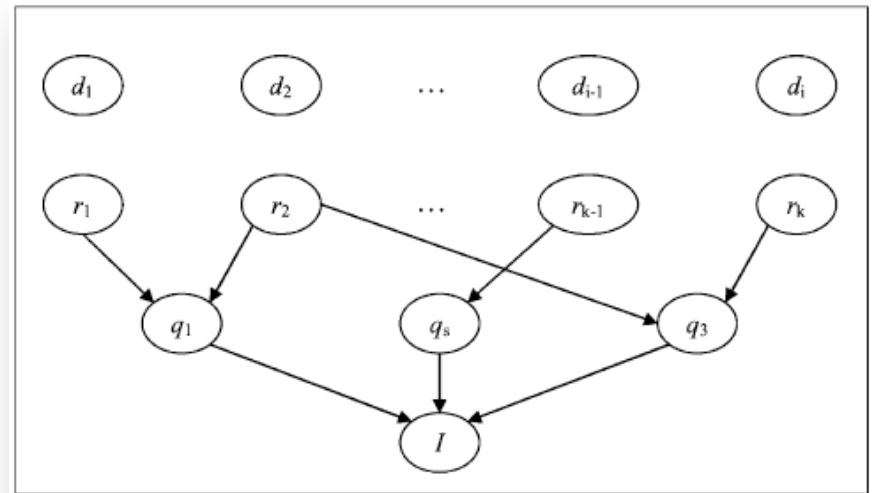
AI-based approaches

Bayesian Networks

- Metzler and Croft (2004)
 - Indri search engine based on InQuery

- Inference network

- Document
- Representation (term, phrases)
- Query
- Information Need



- Calculates probability of each document from the network