Applied Databases

Lecture 10
Full-Text Search

Sebastian Maneth

University of Edinburgh - February 16th, 2017

Outline

1. Text Search
2. Ranking & Similarity Measures
3. Inverted Files
4. Lucene (outlook)

Extra Reading Material

→ Please check course web page.

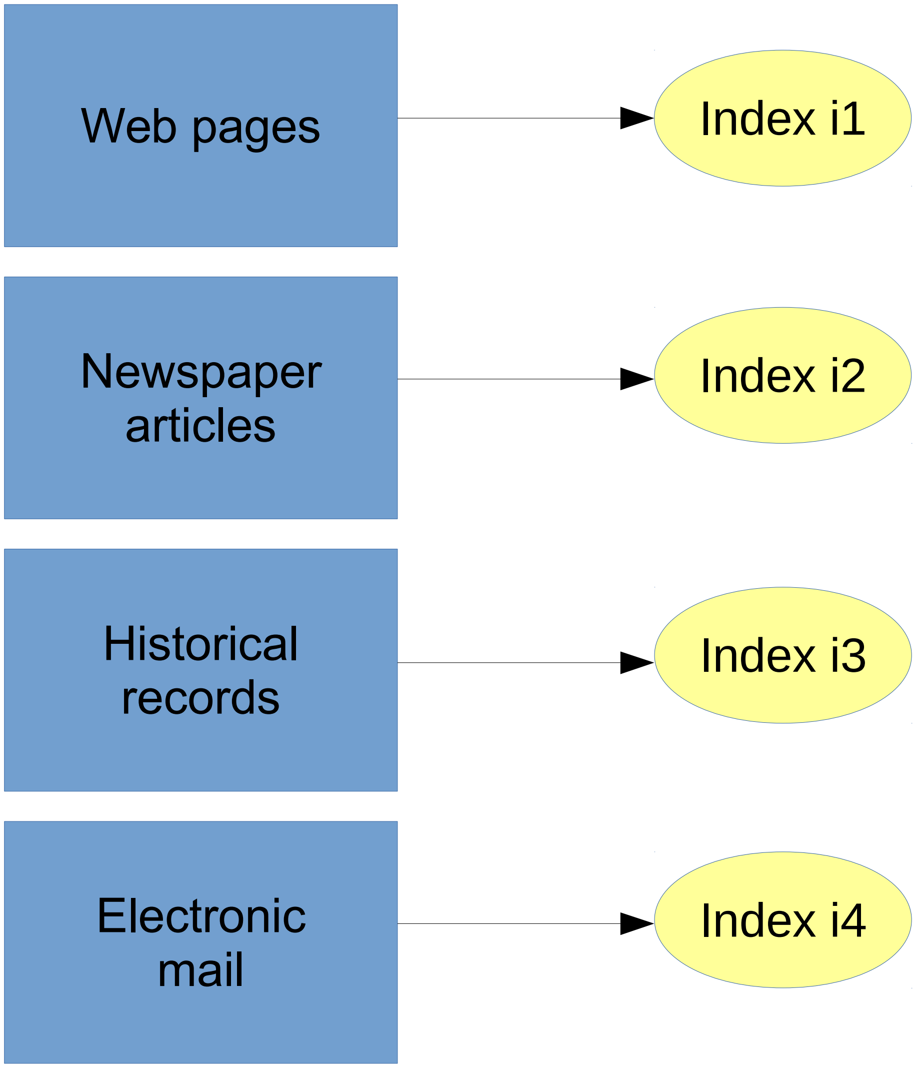
Most of this lecture based on this **article** (PDF linked on course web page)

Zobel, Justin and Moffat, Alistair,
Inverted files for text search engines.
ACM Comput. Surv. 38(2) (2006)

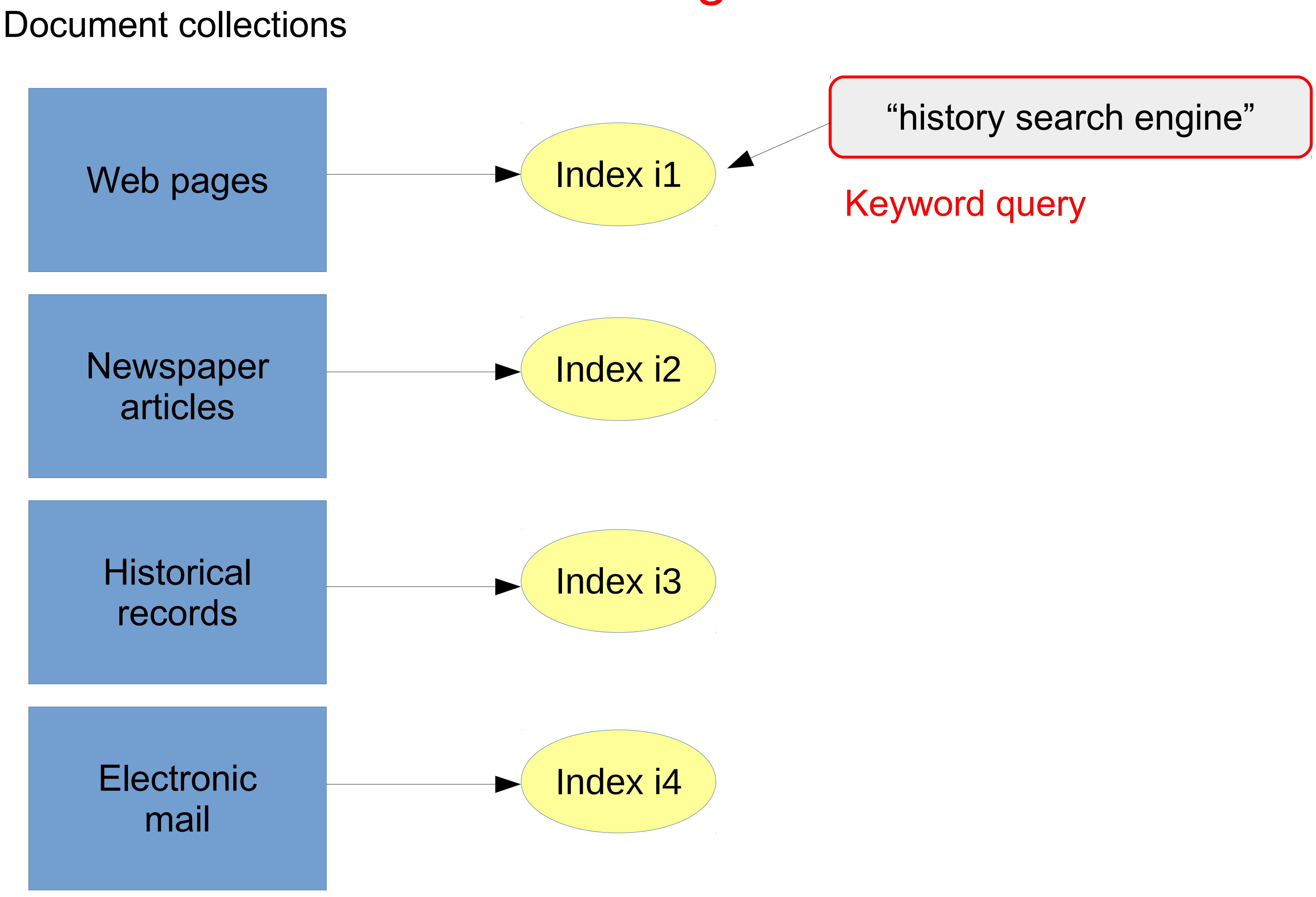
Good read!

Search Engines

Document collections

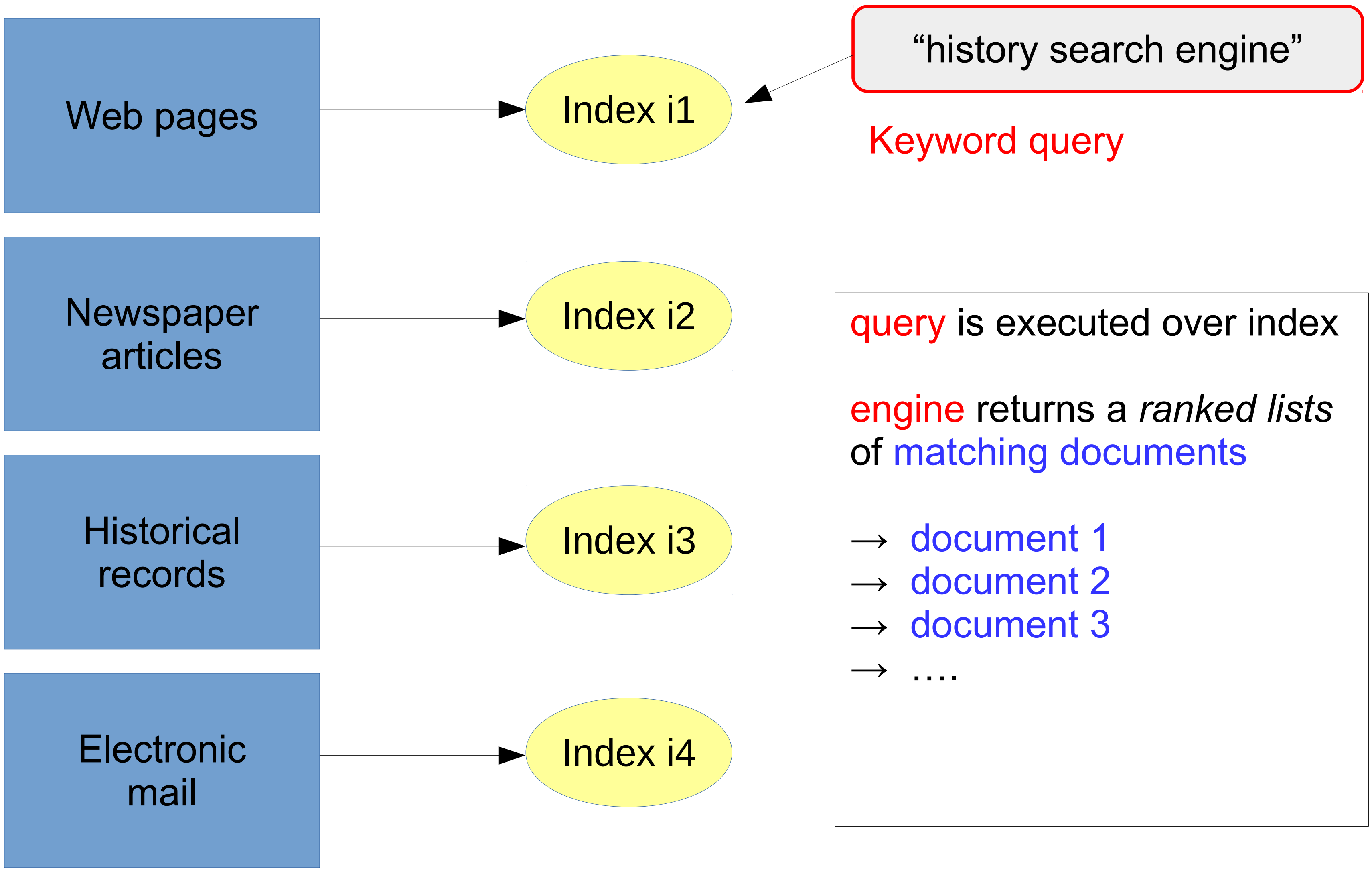


Search Engines



Search Engines

Document collections



1. Text Search

RDBMS search (e.g. SQL)

→ DB system must answer arbitrarily complex queries

→ a match is a tuple that meets a specified logical condition

→ DB systems returns all matching tuples

→ each tuple has a unique access key; may search over that key

Text search

→ most queries are simple lists of terms or phrases

→ a match is a document that is appropriate to the query wrt statistical heuristics (it may not even contain all query terms!)

→ search engine returns fixed number of matches ranked by their statistical similarity

→ there may be millions of documents with non-zero similarity

1. Text Search

Due to these “cultural” differences, the respective research communities of

→ databases

→ information retrieval

have remained separate for many decades (and continue to do so)

For the same reason, we use *separate products* for combined search (→ **Assignments 1 & 2** over ebay data):

→ MySQL
→ Apache Lucene

1. Text Search

→ databases

→ information retrieval

Question for you

→ what is the difference between "data" and "information"?

1. Text Search

Challenges

→ query term may occur in many documents
→ each document may contain many terms

New
→ representations for text indexes
→ index construction techniques
→ algorithms for evaluation of text queries

} crucial for rapid response of major Web Search Engines (e.g. Google or Yahoo)

→ compression and
→ careful organization

} reduction of
– index sizes
– time
– disk traffic during query evaluation

1. Text Search

Search Engine = tool to find documents from a collection that are good matches to a user query

Collections are, e.g., web pages, news articles, emails, etc.

Collections vary dramatically in size

→ 10 years of research papers by a research (plain text)
ca. 10 megabytes

→ 10 years of emails of the researcher
ca. 100 megabytes

→ books in a small university library
ca. 100 gigabytes

→ complete text of the web (year 2006)
ca. 100 terabytes

(in 2014, Google has indexed 200TB, which is claimed to be only 0.4% of the Web)

1. Text Search

Search Engine = tool to find documents from a collection that are good matches to a user query

Most text querying done
→ *by content*
→ satisfies an *information need*

A document matches an information need,
if the user perceives it to be relevant.

→ relevance is inexact!
→ a document may be relevant, but contain none of the query terms or irrelevant, even though it contains all the query terms.

A system is effective, if a good proportion of the first k search results are relevant.

1. Text Search

→ bag-of-words queries

big old house

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

Fig. 1. The Keeper database. It consists of six one-line documents.

→ docs 2 and 3 contain all query terms

→ docs 1 and 4 contain "old"

→ only doc 2 contains the *phrase* "big old house"

1. Text Search

Parsing method for extracting terms from text:

→ should HTML markup be indexed?
→ or terms that appear within markup?
→ hyphenated words, considered as one or two words?

More fundamentally:

→ stemming? (= remove variant endings of a word)
→ casefolding? (= convert to lowercase)
→ stopping? (= remove common / functions words, e.g. "the")

1. Text Search

→ stemming? (= remove variant endings of a word)
→ casefolding? (= concert to lowercase)
→ stopping? (= remove common or functions words, e.g. “the”)

```
The old night keeper keeps the keep in the town
In the big old house in the big old gown.
The house in the town had the big old keep
Where the old night keeper never did sleep.
The night keeper keeps the keep in the night
And keeps in the dark and sleeps in the light.
```

term parser

with casefolding (sorted vocabulary)

```
and big dark did gown had house in keep keeper keeps light
never night old sleep sleeps the town where
```

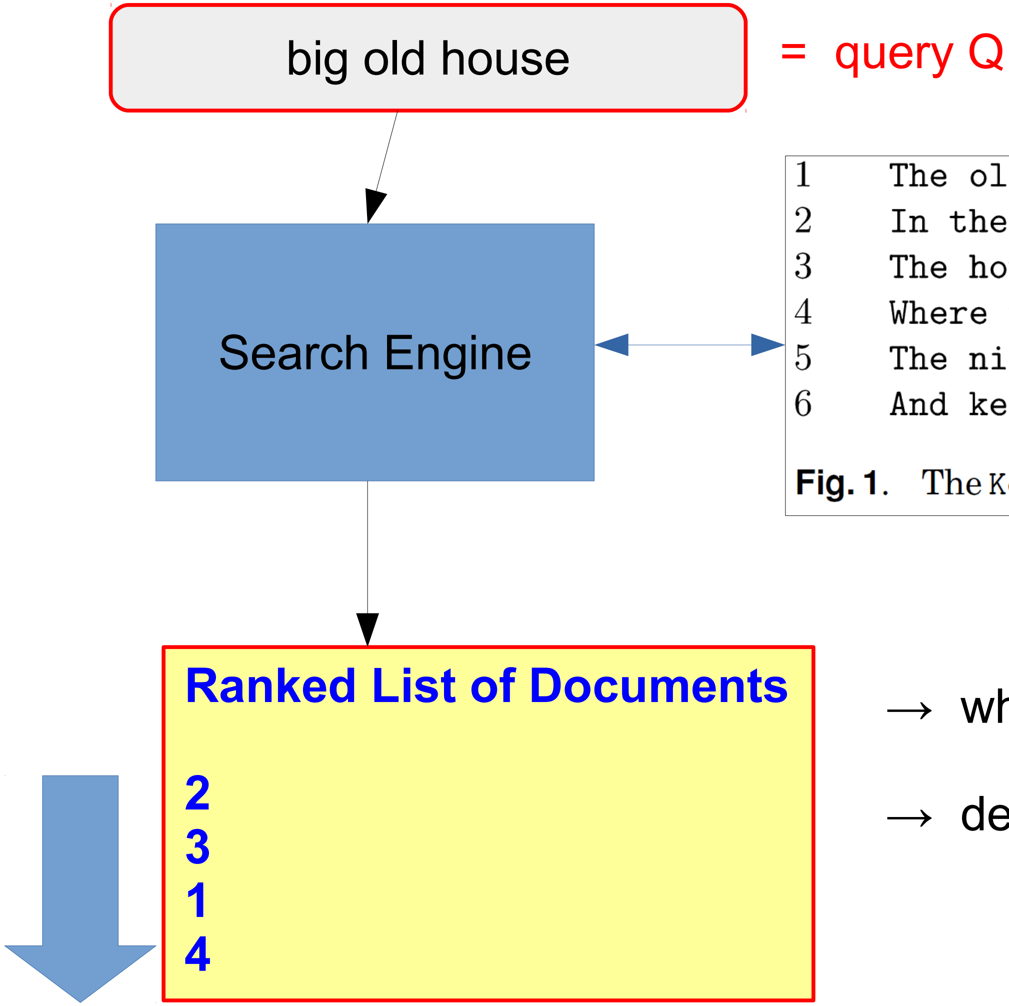
with stemming

```
and big dark did gown had house in keep light never night old
sleep the town where
```

with stopping

```
big dark gown house keep light night old sleep town
```

2. Ranking and Similarity Measures




```
1    The old night keeper keeps the keep in the town
2    In the big old house in the big old gown.
3    The house in the town had the big old keep
4    Where the old night keeper never did sleep.
5    The night keeper keeps the keep in the night
6    And keeps in the dark and sleeps in the light.
```

Fig. 1. The Keeper database. It consists of six one-line documents.

→ which document is *closest* to query?

→ define **similarity measure S**(Q, D)

query Q

document D

2. Similarity Measures

→ how to define a good **similarity measure**?

(1) give higher score if many query terms appear in the document (many times)

2. Similarity Measures

→ how to define a good **similarity measure**?

(1) give higher score if many query terms appear in the document (many times)

(2) give less weight to terms that appear in many documents

(3) give more weight to terms that appear many times in a document

(4) give less weight to documents that contain many terms.

Term Frequency (TF)
f(D,T) = how many times does term T appear in document D?

Document Frequence (DF)
f(T) = in how many documents of the collection does term T appear?

2. Similarity Measures

Term Frequency (TF)
f(D,T) = how many times does term T appear in document D?

Document Frequence (DF)
f(T) = in how many documents of the collection does term T appear?

Inverse Document Frequence (IDF)
1 / f(T)

TF * IDF = f(D,T) / f(T)

→ e.g. "old" appears in 4 documents (out of 6)

f(1,"old") = 1, thus	TF*IDF	=	1 / 4
f(2,"old") = 2, thus	TF*IDF	=	2 / 4
f(3,"old") = 1, thus	TF*IDF	=	1 / 4
f(4,"old") = 1, thus	TF*IDF	=	1 / 4

2. Similarity Measures

Term Frequency (TF)
f(D,T) = how many times does term T appear in document D?

Document Frequence (DF)
f(T) = in how many documents of the collection does term T appear?

Inverse Document Frequence (IDF)
1 / f(T)

TF * IDF = f(D,T) / f(T)

→ e.g. "old" appears in 4 documents (out of 6)

f(1,"old") = 1, thus TF*IDF = 1 / 4
f(2,"old") = 2, thus TF*IDF = 2 / 4
f(3,"old") = 1, thus TF*IDF = 1 / 4
f(4,"old") = 1, thus TF*IDF = 1 / 4

could be 300 appearances!

2. Similarity Measures

Term Frequency (TF)
f(D,T) = how many times does term T appear in document D?

Document Frequence (DF)
f(T) = in how many documents of the collection does term T appear?

Inverse Document Frequence (IDF)
1 / f(T)

ignored (so far):
N = number of documents
in the collection

TF * IDF = f(D,T) / f(T)

→ e.g. "old" appears in 4 documents (out of 6)

f(1,"old") = 1, thus TF*IDF = 1 / 4
f(2,"old") = 2, thus TF*IDF = 2 / 4
f(3,"old") = 1, thus TF*IDF = 1 / 4
f(4,"old") = 1, thus TF*IDF = 1 / 4

2. Similarity Measures

Term Frequency (TF) (non-scaled) scaled: 1 + ln(f(D,T))
f(D,T) = how many times does term T appear in document D?

Document Frequence (DF)
f(T) = in how many documents of the collection does term T appear?

Inverse Document Frequence (IDF) (non-scaled)
1 / f(T)

TF * IDF = f(D,T) / f(T)

ignored (so far):
N = number of documents in the collection

→ e.g. "old" appears in 4 documents (out of 6)
f(1,"old") = 1, thus TF*IDF = 1 / 4
f(2,"old") = 2, thus TF*IDF = 2 / 4

IDF = ln (1 + N / DF) – "scaled"

Thus, TF*IDF for "old" and doc1: (1+ln(1))*ln(1+ 6/4) = 0.916
"old" and doc2: (1+ln(2))*ln(1+ 6/4) = 1.551

Example (patents)

Here: IDF = log(N/DF)

Term	TF(doc1)	TF(doc2)	TF(doc3)	DF	IDF
method	4,250	**3,400**	5,100	850	**0.27**
the	50,000	43,000	55,000	1,000	**0.00**
water	7,600	**4,000**	2,000	400	**0.54**
bioreactor	600	0	25	25	**1.6**

Here: TF is not scaled

term	TF-IDF(doc1)	TF-IDF(doc2)	TF-IDF(doc3)
method	1148	**918**	1377
the	0	0	0
water	4104	**2160**	1080
bioreactor	960	0	40

inverse document frequency influences the TF-IDF value:

→ "method" occurs nearly as often as "water" in doc2 but TF-IDF value of "water" is more than double that of “method”

→ a query "method bioreactor" would assign doc1 a score of 0.15 and doc3 a score of 0.04.

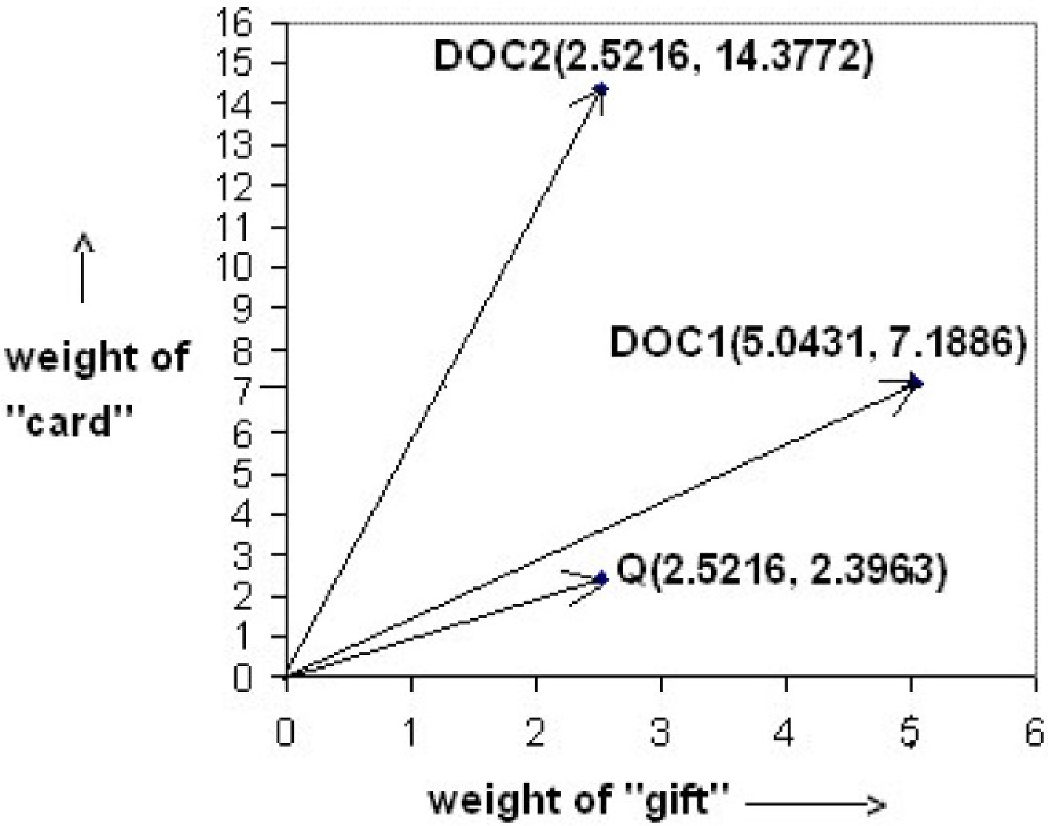
2. Similarity Measures

Given a query (T1, T2, .., Tk), compute for each document D the vector

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

<TFIDF(T1, D), …, TFIDF(Tk, D)>

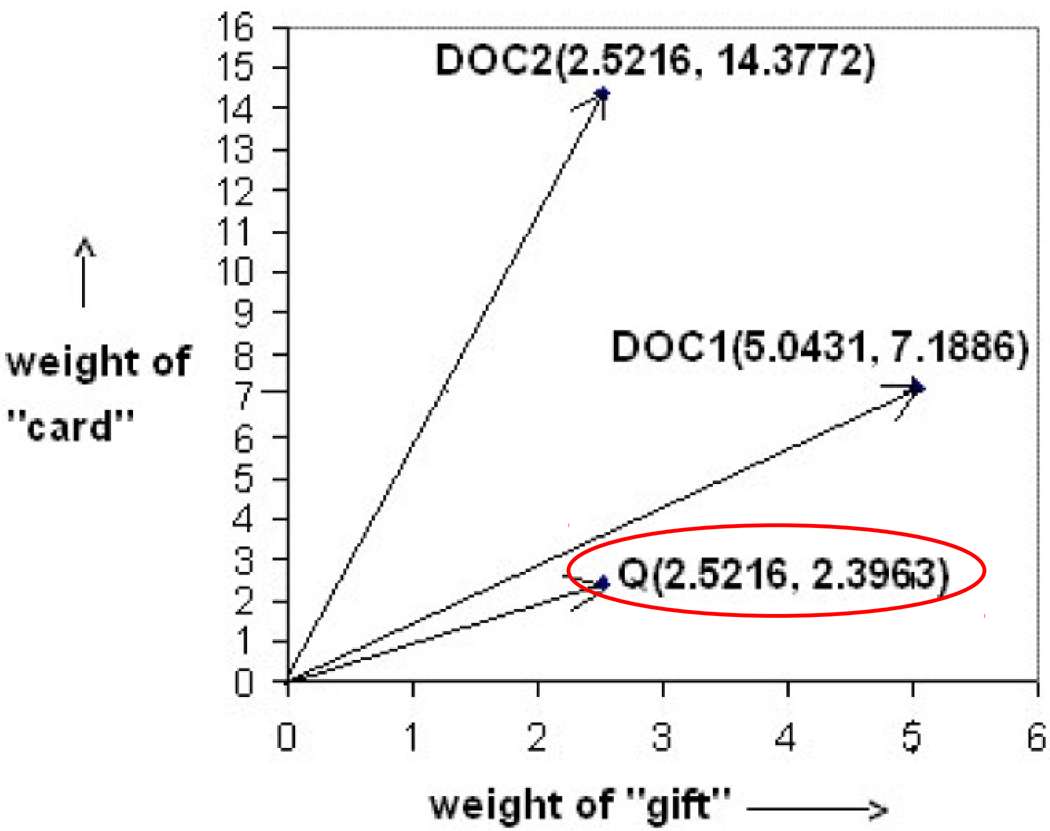


2. Similarity Measures

Given a query (T1, T2, .., Tk), compute for each document D the vector

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

<TFIDF(T1, D), …, TFIDF(Tk, D) >

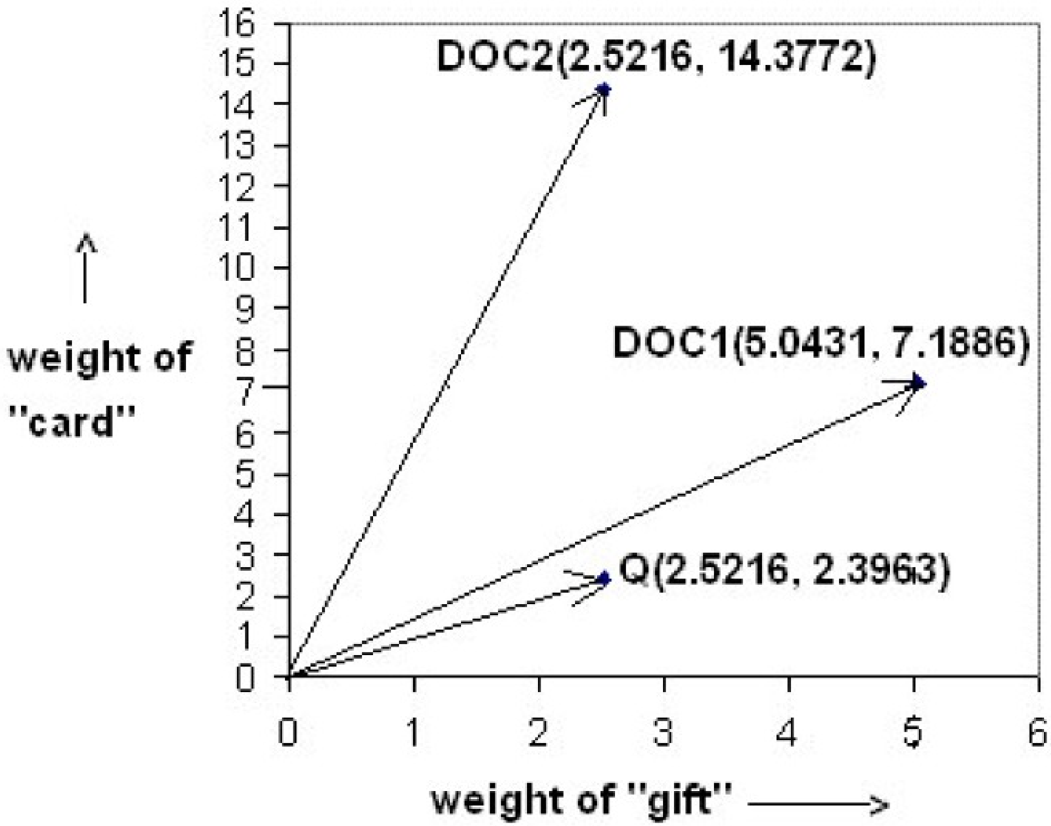


for query q compute vector
< TFIDF(T1, q), …, TFIDF(Tk, q)>

2. Similarity Measures

→ **angle** between DOC-k and Q determines similarity
(length of vector not important)

→ "relative closeness" of term weights



→ the closer angle is to zero, the more similar the documents

→ if angle is >=90 degrees, then documents have no words in common

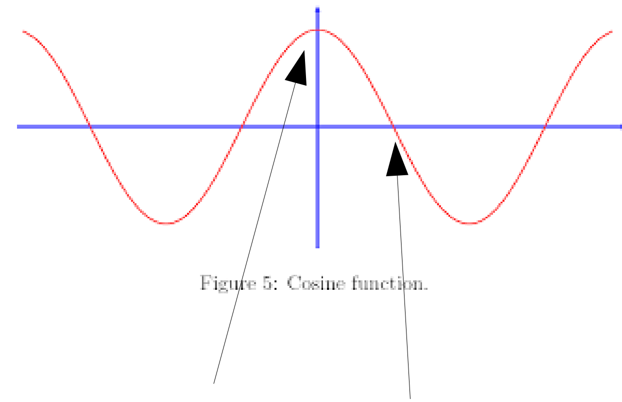
→ DOC1 and Q are very similar!

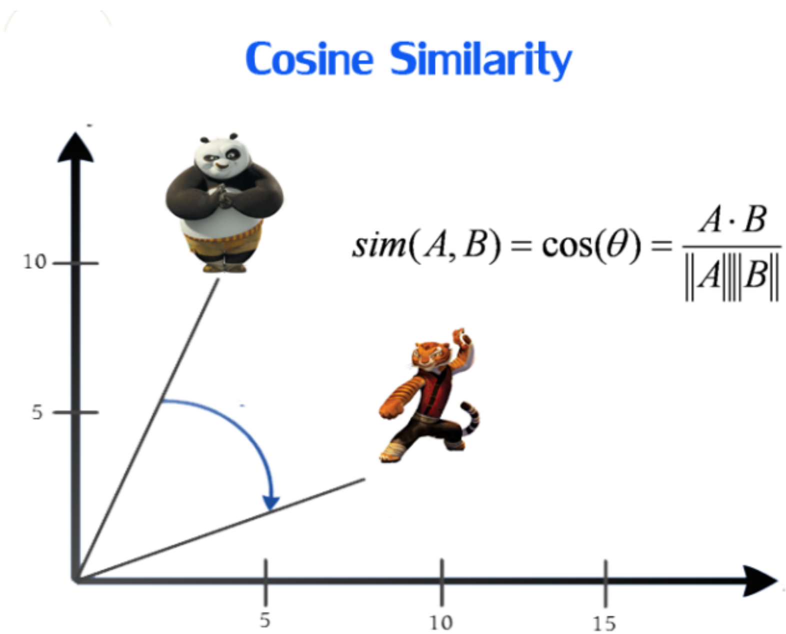
2. Similarity Measures

Given a query (T1, T2, .., Tk), compute for each document D the vector

<TFIDF(T1, D), …, TFIDF(Tk, D) >

consider cosine similarity between such vector A and vector B for the query





cosine similarity:

cos(angle between A and B)

→ equals "1" if angle is zero (vectors have same direction)

→ equals "0" if orthogonal (90 degree) (means: no words in common)

2. Similarity Measures

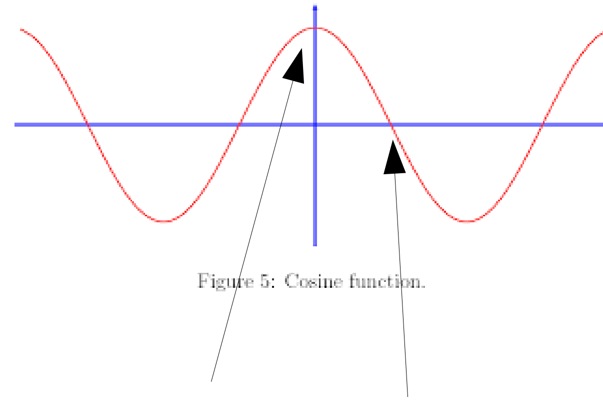


Figure 5: Cosine function.

Given a query (T1, T2, .., Tk), compute for each document D the vector

<TFIDF(T1, D), …, TFIDF(Tk, D) >

consider cosine similarity between such vector A and vector B for the query

$$\text{Similarity(A,B)} = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

cosine similarity:

cos(angle between A and B)

→ equals "1" if angle is zero (vectors have same direction)

→ equals "0" if orthogonal (90 degr) (means: no words in common)

3. Inverted Indexes / Files

Fast query evaluation makes use of an Index.

3. Inverted Indexes / Files

Fast query evaluation makes use of an Index.

Index = datastructure that maps terms to documents containing them

3. Inverted Indexes / Files

Fast query evaluation makes use of an Index.

Index = datastructure that maps terms to documents containing them

E.g. consider pages of a book as "documents"

Book index: maps words to pages

A concordance is an alphabetical list of the principal words used in a book or body of work, listing every instance of each word with its immediate context. Because of the time, difficulty, and expense involved in creating a concordance in the pre-computer era, only works of special importance, such as the Vedas [1] Bible, Qur'an or the works of Shakespeare or classical Latin had concordances prepared for them.

The first Bible concordance, for the Vulgate Bible, was compiled by Hugh of St Cher (d.**1262**), who employed 500 monks to assist him. In 1448 Rabbi Mordecai Nathan completed a concordance to the Hebrew Bible. It took him ten years.

3. Inverted Indexes / Files

Inverted File = for each distinct word T, contains
→ f(T) (#documents that contain T)
→ pointer to the corresponding inverted list

} vocabulary

Inverted List of T = pairs < D, f(D,T) >
Listing each document D that contains T,
with number f(D,T) of occurrences of T in D

Thumb rule for effective retrieval:

→ index **all terms**, even stop words, numbers, etc.

1 The old night keeper keeps the keep in the town
2 In the big old house in the big old gown.
3 The house in the town had the big old keep
4 Where the old night keeper never did sleep.
5 The night keeper keeps the keep in the night
6 And keeps in the dark and sleeps in the light.

+ casefolding

term t	f_t	Inverted list for t
and	1	$\langle 6,2\rangle$
big	2	$\langle 2,2\rangle$ $\langle 3,1\rangle$
dark	1	$\langle 6,1\rangle$
did	1	$\langle 4,1\rangle$
gown	1	$\langle 2,1\rangle$
had	1	$\langle 3,1\rangle$
house	2	$\langle 2,1\rangle$ $\langle 3,1\rangle$
in	5	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$ $\langle 6,2\rangle$
keep	3	$\langle 1,1\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$
keeper	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,1\rangle$
keeps	3	$\langle 1,1\rangle$ $\langle 5,1\rangle$ $\langle 6,1\rangle$
light	1	$\langle 6,1\rangle$
never	1	$\langle 4,1\rangle$
night	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,2\rangle$
old	4	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 4,1\rangle$
sleep	1	$\langle 4,1\rangle$
sleeps	1	$\langle 6,1\rangle$
the	6	$\langle 1,3\rangle$ $\langle 2,2\rangle$ $\langle 3,3\rangle$ $\langle 4,1\rangle$ $\langle 5,3\rangle$ $\langle 6,2\rangle$
town	2	$\langle 1,1\rangle$ $\langle 3,1\rangle$
where	1	$\langle 4,1\rangle$

doc-6 contains “and” 2 times

only 1 document contains “and”

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

term t	f_t	Inverted list for t
and	1	$\langle 6,2\rangle$
big	2	$\langle 2,2\rangle$ $\langle 3,1\rangle$
dark	1	$\langle 6,1\rangle$
did	1	$\langle 4,1\rangle$
gown	1	$\langle 2,1\rangle$
had	1	$\langle 3,1\rangle$
house	2	$\langle 2,1\rangle$ $\langle 3,1\rangle$
in	5	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$ $\langle 6,2\rangle$
keep	3	$\langle 1,1\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$
keeper	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,1\rangle$
keeps	3	$\langle 1,1\rangle$ $\langle 5,1\rangle$ $\langle 6,1\rangle$
light	1	$\langle 6,1\rangle$
never	1	$\langle 4,1\rangle$
night	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,2\rangle$
old	4	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 4,1\rangle$
sleep	1	$\langle 4,1\rangle$
sleeps	1	$\langle 6,1\rangle$
the	6	$\langle 1,3\rangle$ $\langle 2,2\rangle$ $\langle 3,3\rangle$ $\langle 4,1\rangle$ $\langle 5,3\rangle$ $\langle 6,2\rangle$
town	2	$\langle 1,1\rangle$ $\langle 3,1\rangle$
where	1	$\langle 4,1\rangle$

w(d,t) = 1 + ln f(D,T)

$$W_d = \sum_t w_{d,t}^2$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

→ how do doc-2 and doc-4 differ?

→ doc-4 is more “specific”

```
1   The old night keeper keeps the keep in the town
2   In the big old house in the big old gown.
3   The house in the town had the big old keep
4   Where the old night keeper never did sleep.
5   The night keeper keeps the keep in the night
6   And keeps in the dark and sleeps in the light.
```

+ casefolding

term t	f_t	Inverted list for t
and	1	$\langle 6, 2\rangle$
big	2	$\langle 2, 2\rangle$ $\langle 3, 1\rangle$
dark	1	$\langle 6, 1\rangle$
did	1	$\langle 4, 1\rangle$
gown	1	$\langle 2, 1\rangle$
had	1	$\langle 3, 1\rangle$
house	2	$\langle 2, 1\rangle$ $\langle 3, 1\rangle$
in	5	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 5, 1\rangle$ $\langle 6, 2\rangle$
keep	3	$\langle 1, 1\rangle$ $\langle 3, 1\rangle$ $\langle 5, 1\rangle$
keeper	3	$\langle 1, 1\rangle$ $\langle 4, 1\rangle$ $\langle 5, 1\rangle$
keeps	3	$\langle 1, 1\rangle$ $\langle 5, 1\rangle$ $\langle 6, 1\rangle$
light	1	$\langle 6, 1\rangle$
never	1	$\langle 4, 1\rangle$
night	3	$\langle 1, 1\rangle$ $\langle 4, 1\rangle$ $\langle 5, 2\rangle$
old	4	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 4, 1\rangle$
sleep	1	$\langle 4, 1\rangle$
sleeps	1	$\langle 6, 1\rangle$
the	6	$\langle 1, 3\rangle$ $\langle 2, 2\rangle$ $\langle 3, 3\rangle$ $\langle 4, 1\rangle$ $\langle 5, 3\rangle$ $\langle 6, 2\rangle$
town	2	$\langle 1, 1\rangle$ $\langle 3, 1\rangle$
where	1	$\langle 4, 1\rangle$

$$w(d,t) = 1 + \ln f(D,T)$$

$$W_d = \sum_t w_{d,t}^2$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

4*(1+ ln 2)^2 + 2 = 13.4666

```
1    The old night keeper keeps the keep in the town
2    In the big old house in the big old gown.
3    The house in the town had the big old keep
4    Where the old night keeper never did sleep.
5    The night keeper keeps the keep in the night
6    And keeps in the dark and sleeps in the light.
```

+ casefolding

term t	f_t	Inverted list for t
and	1	$\langle 6,2\rangle$
big	2	$\langle 2,2\rangle$ $\langle 3,1\rangle$
dark	1	$\langle 6,1\rangle$
did	1	$\langle 4,1\rangle$
gown	1	$\langle 2,1\rangle$
had	1	$\langle 3,1\rangle$
house	2	$\langle 2,1\rangle$ $\langle 3,1\rangle$
in	5	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$ $\langle 6,2\rangle$
keep	3	$\langle 1,1\rangle$ $\langle 3,1\rangle$ $\langle 5,1\rangle$
keeper	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,1\rangle$
keeps	3	$\langle 1,1\rangle$ $\langle 5,1\rangle$ $\langle 6,1\rangle$
light	1	$\langle 6,1\rangle$
never	1	$\langle 4,1\rangle$
night	3	$\langle 1,1\rangle$ $\langle 4,1\rangle$ $\langle 5,2\rangle$
old	4	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 4,1\rangle$
sleep	1	$\langle 4,1\rangle$
sleeps	1	$\langle 6,1\rangle$
the	6	$\langle 1,3\rangle$ $\langle 2,2\rangle$ $\langle 3,3\rangle$ $\langle 4,1\rangle$ $\langle 5,3\rangle$ $\langle 6,2\rangle$
town	2	$\langle 1,1\rangle$ $\langle 3,1\rangle$
where	1	$\langle 4,1\rangle$

w(d,t) = 1 + ln f(D,T)

W_d = $\sum_t w_{d,t}^2$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

4*(1+ ln 2)^2 + 2 = 13.4666

8*(1+ ln 1)^2 = 8

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

query score S(q,d) for document d on query q,
from [Zobel, Moffat 2006]

(incorporates cosine-simularity and TF*IDFT)

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = old

→ compute **score of S(q,d)** of **query q** on documents 3 and 4:

old	4	$\langle 1,1\rangle \ \langle 2,2\rangle \ \langle 3,1\rangle \ \langle 4,1\rangle$

inverted file entry for "old"

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = old

→ compute **score of S(q,d)** of **query q** on documents 3 and 4:

S(q,doc-3) = w(doc-3, “old”) * w(q, “old”) / 11.4 = (1 + ln(1)) * ln(1 + 6/4) / 11.4
= **0.0804**

old	4	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 4,1\rangle$

inverted file entry for “old”

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = old

→ compute **score of S(q,d)** of **query q** on documents 3 and 4:

S(q,doc-3) = w(doc-3, "old") * w(q, "old") / 11.4 = (1 + ln(1)) * ln(1 + 6/4) / 11.4
= **0.0804**

S(q,doc-4) = w(doc-4, "old") * w(q, "old") / 8 = (1 + ln(1)) * ln(1 + 6/4) / 8
= **0.1145**

old	4	$\langle 1,1\rangle\ \langle 2,2\rangle\ \langle 3,1\rangle\ \langle 4,1\rangle$

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = old

→ compute **score of S(q,d)**
of **query q** on documents 3 and 4:

S(q,doc-3) = w(doc-3, “old”) * w(q, “old”) / 11.4 = (1 + ln(1)) * ln(1 + 6/4) / 11.4
= **0.0804**

S(q,doc-4) = w(doc-4, “old”) * w(q, “old”) / 8 = (1 + ln(1)) * ln(1 + 6/4) / 8
= **0.1145**

→ doc-4 has higher score because of lower W_d value! (it is more 'specific')

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = big old house

→ want to compute **score of S(q,d)** of **query q** on document 2

→ need to compute:

(w(**doc-2**,"big") * w(**q**,"big")
+ w(**doc-2**,"old") * w(**q**,"old")
+ w(**doc-2**,"house") * w(**q**,"house")) / 13.5

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

→ want to compute **score of S(q,d)** of **query q** on document 2

→ need to compute:

(w(**doc-2**,"big") * w(**q**,"big")
+ w(**doc-2**,"old") * w(**q**,"old")
+ w(**doc-2**,"house") * w(**q**,"house")) / 13.5

All we need are the **inverted file** entries for "big", "old", and "house"!

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

w(q,"big") = ln(1 + 6 / 2) = ln(4)

w(doc-2, "big") = 1 + ln(f(doc-2, "big")) = 1+ ln(2)

big	2	$\langle 2, 2\rangle$ $\langle 3, 1\rangle$	old	4	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 4, 1\rangle$	house	2	$\langle 2, 1\rangle$ $\langle 3, 1\rangle$

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

w(q,"big") = ln(1 + 6 / 2) = ln(4)

w(doc-2, "big") = 1 + ln(f(doc-2, "big")) = 1+ ln(2)

S(q,doc-2) = w(doc-2, "big") * w(q, "big") / W(doc-2) + .. = (1 + ln(2)) * ln(4) / 13.5
= **2.3472 / 13.5** + ..

big	2	$\langle 2, 2\rangle$ $\langle 3, 1\rangle$	old	4	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 4, 1\rangle$
house	2	$\langle 2, 1\rangle$ $\langle 3, 1\rangle$			

1 The old night keeper keeps the keep in the town
2 In the big old house in the big old gown.
3 The house in the town had the big old keep
4 Where the old night keeper never did sleep.
5 The night keeper keeps the keep in the night
6 And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

w(q,"old") = ln(1 + 6 / 4) = ln(2.5)

w(doc-2, "old") = 1 + ln(f(doc-2, "old")) = 1+ ln(2)

S(q,doc-2) = (2.3472 + ln(2.5) * (1 + ln(2))) / 13.5 = **(2.3472 + 1.5514) / 13.5**

big	2	$\langle 2, 2\rangle$ $\langle 3, 1\rangle$
old	4	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 4, 1\rangle$
house	2	$\langle 2, 1\rangle$ $\langle 3, 1\rangle$

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

w(q,"house") = ln(1 + 6 / 2) = ln(4)

w(doc-2, "house") = 1 + ln(f(doc-2, "house")) = 1+ ln(1) = 1

S(q,doc-2) = (3.8986 + ln(4) * 1) / 13.5 = (3.8986 + 1.3863) / 13.5 = **0.3915**

big 2	⟨2, 2⟩ ⟨3, 1⟩	old 4	⟨1, 1⟩ ⟨2, 2⟩ ⟨3, 1⟩ ⟨4, 1⟩	house 2	⟨2, 1⟩ ⟨3, 1⟩

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = big old house

w(q,"big") = ln(1 + 6 / 2) = ln(4)

w(doc-3, "big") = 1 + ln(f(doc-3, "big")) = 1+ ln(1) = 1

S(q,doc-3) = w(doc-3, "big") * w(q, "big") / W(doc-3) + .. = ln(4) / 11.4 =

1.3863 / 11.4

big	2	⟨2, 2⟩ ⟨3, 1⟩	old	4	⟨1, 1⟩ ⟨2, 2⟩ ⟨3, 1⟩ ⟨4, 1⟩	house	2	⟨2, 1⟩ ⟨3, 1⟩

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = big old house

w(q,"old") = ln(1 + 6 / 4) = ln(2.5)

w(doc-3, "old") = 1 + ln(f(doc-3, "old")) = 1+ ln(1) = 1

S(q,doc-3) = (**1.3862** + **ln(2.5)** + ..) / 11.4 = **(1.3863 + 0.9163) / 11.4**

big	2	$\langle 2,2\rangle$ $\langle 3,1\rangle$
old	4	$\langle 1,1\rangle$ $\langle 2,2\rangle$ $\langle 3,1\rangle$ $\langle 4,1\rangle$
house	2	$\langle 2,1\rangle$ $\langle 3,1\rangle$

1	The old night keeper keeps the keep in the town
2	In the big old house in the big old gown.
3	The house in the town had the big old keep
4	Where the old night keeper never did sleep.
5	The night keeper keeps the keep in the night
6	And keeps in the dark and sleeps in the light.

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right)$$

$$w_{d,t} = 1 + \ln f_{d,t}$$

$$W_d = \sum_t w_{d,t}^2$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

q = big old house

w(q,"old") = ln(1 + 6 / 2) = ln(4)

w(doc-3, "old") = 1 + ln(f(doc-3, "old")) = 1+ ln(1) = 1

S(q,doc-3) = **(1.3863 + 0.9163 + ln(4)) / 11.4 = 0.3236**

big	2	$\langle 2, 2\rangle$ $\langle 3, 1\rangle$
old	4	$\langle 1, 1\rangle$ $\langle 2, 2\rangle$ $\langle 3, 1\rangle$ $\langle 4, 1\rangle$
house	2	$\langle 2, 1\rangle$ $\langle 3, 1\rangle$

```
1     The old night keeper keeps the keep in the town
2     In the big old house in the big old gown.
3     The house in the town had the big old keep
4     Where the old night keeper never did sleep.
5     The night keeper keeps the keep in the night
6     And keeps in the dark and sleeps in the light.
```

+ casefolding

$$w_{q,t} = \ln\left(1 + \frac{N}{f_t}\right) \qquad w_{d,t} = 1 + \ln f_{d,t}$$

d	1	2	3	4	5	6
W_d	11.4	13.5	11.4	8.0	11.3	12.6

$$W_d = \sqrt{\sum_t w_{d,t}^2}$$

$$S_{q,d} = \frac{\sum_t w_{d,t} \cdot w_{q,t}}{W_d}.$$

q = big old house

S(q,doc-2) = (3.8986 + 1.3863) / 13.5 = **0.3915**

S(q,doc-3) = **(1.3863 + 0.9163 + ln(4)) / 11.4 = 0.3236**

→ doc-2 is **ranked higher** (more relevant) than doc-3 for query q

big	2	⟨2, 2⟩ ⟨3, 1⟩	old	4	⟨1, 1⟩ ⟨2, 2⟩ ⟨3, 1⟩ ⟨4, 1⟩	house	2	⟨2, 1⟩ ⟨3, 1⟩

3. Inverted Indexes / Files

→ inverted lists are stored contiguously

→ vocabulary stored in simple extensible structure (e.g., B-tree)
(may be preprocessed by stemming and stopping)

→ inverted lists consist of doc numbers with #occurrences
(possibly augmented by word positions)

→ ranking involves a set of accumulators and
term-by-term processing of inverted lists

4. Lucene (outlook)

Lucene allows you to take care of everything mentioned today:

4. Lucene (outlook)

Lucene allows you to take care of everything mentioned today:

→ you can choose different Analyzers to do
- casefolding
- stemming (wrt a given language)
- stopping (wrt a given language)

→ you can insert documents into a collection and let Lucene generate inverted files for you (= “indexing” – very efficient!)

4. Lucene (outlook)

Lucene allows you to take care of everything mentioned today:

→ you can choose different Analyzers to do
- casefolding
- stemming (wrt a given language)
- stopping (wrt a given language)

→ you can insert documents into a collection and let Lucene generate inverted files for you (= "indexing" – very efficient!)

→ you can then (very efficiently) retrieve the k top-most relevant documents in your collection!

→ ranking function is a bit more sophisticated

$$score(q,d) = \sum [tf(t_d) \times idf(t) \times boost(t.field_d) \times lengthNorm(t.field_d)] \times coord(q,d) \times qNorm(q)$$

Questions

1) sizes of inverted files?

Questions

1) sizes of inverted files?

	NewsWire	Web
Size (gigabytes)	1	100
Documents	400,000	12,000,000
Word occurrences (without markup)	180,000,000	11,000,000,000
Distinct words (after stemming) . . . ,	400,000	16,000,000
per document, totaled	70,000,000	3,500,000,000

Size of Inverted Index for **NewsWire (1 GB): 435 MB**

- **12MB** for 400,000 words, pointers, and counts
- **1.6MB** for 400,000 W(D)-values
- **280MB** for 70,000,000 document identifiers (four bytes each)
- **140MB** for 70,000,000 document frequencies (two bytes each)

Questions

1) sizes of inverted files?

2) limits of inverted files

→ imagine **substring search** (e.g. in DNA strands)

→ number of substrings is quadratic, cannot possibly generate/store them!

Questions

1) sizes of inverted files?

2) limits of inverted files

→ imagine **substring search** (e.g. in DNA strands)

→ number of substrings is quadratic, cannot possibly generate/store them!

SOLUTION:

3) in-memory indexes (for substring search, like DNA)

→ occupy a fraction of 435MB (40% of NewsWire)
→ run much faster :-)

Google's speed = (in-memory + MANY machines)

Questions

1) sizes of inverted files?

2) limits of inverted files

   → imagine **substring search** (e.g. in DNA strands)

   → number of substrings is quadratic, cannot possibly generate/store them!

3) in-memory indexes for substring search

---

4) online substring search (without indexes)

END
Lecture 10