DMML, 26 March 2019

Information Retrieval

Corpus of documents

Information need → query → return matching documents

Boolean document model

Term-document matrix

Compress this as postings list

$$t \to n, [d_1, d_2, \ldots, d_k]$$

Answer boolean queries

$(t_1 \wedge t_2) \vee \neg t_3$

Merge postings list appropriately

Choosing the terms to index

Stop words

Finding a root/canonical form

Stemming

Lemmatization

Information need → query → list of responses

What do we expect form the returned list?

Ranked retrieval

Existing postings cannot distinguish relevance of different docs that match a query

Need some extra information

Logical units of a document – title, author, abstract, body, ...

Books by JK Rowling

Focus on author field vs body

Explicit metadata - structure is given to the indexing algorithm

Maintain separate postings for each structural unit - title, author

Parametric index

Fields vs zones

william.abstract → 11 → 121 → 1441 → 1729

william.title → 2 → 4 → 8 → 16

william.author → 2 → 3 → 5 → 8

Merge these by tagging the posting entries

william → 2.author,2.title → 3.author → 4.title → 5.author

Given a better structured query, search the appropriate parameterized index

What happens if the query is <u>not</u> explicitly structured?

Google : "J K Rowling"

How to decide a ranking in this case

Weighted sum of parametric scores

$$\sum_i g_i s_i$$

0/1 in the appropriate index

weight of this index

$\sum g_i = 1$

query → score for each matching document, use this to rank search

Use regression to learn g_i

Training data:

query	document	relevant?
—	—	—
— —	—	

Another strategy

Consider words "ball", "net", "point"

Occur most frequently in sports articles

Move away from Boolean model —
frequency of occurrence of t in d
is also important

Term frequency

$tf_{t,d}$ no. of times t appears in doc. d

Frequency vs rarity (recall stop words)

Document frequency :

N documents

n_t # of docs where t appears

$$\frac{n_t}{N}$$

A term is more useful as an indicator if it is less frequent

Inverse document frequency $\log \frac{N}{n_t}$

$$idf_t = \log \frac{N}{n_t}$$

Score of t in d is $tf_{t,d} \times idf_t$

TF-IDF score

query $= \{t_1, t_2, t_3\}$

Given d: TF-IDF score in d for t_1, t_2, t_3

Posting

$t_1 \longrightarrow \{d_1 : s_1,\ d_2 : s_2,\ \ldots\ d_K : s_K\}$

$n_{t_j} \cdot \boxed{idf\ t_1}$

independent of d

Instead

$$t_1 \rightarrow idf_{t_1}, \{d_1 : n_1, d_2 : n_2, \ldots, d_k : n_k\}$$

TF·IDF score

Given q & TF-IDF scores for each d, rank according to these scores

Drawback

Duplicate the content of a document 1000 times

TF grows by factor of 1000 !

More sophisticated model

Think of column for d in term-doc matrix $\rightarrow$ vector

Vector space model

Each doc. is a vector over terms,
entry i is TF-IDF score for t_i

d_2 is 1000 copies of d_1

$$d_2 \downarrow v_2 \qquad d_1 \downarrow v_1$$

$$v_2 = 1000 \cdot v_1$$

Same direction
Magnitude differs

Direction is relevant quantity to compare docs

$$V_1 \cdot V_2 = |V_1|\,|V_2| \underline{\underline{\cos\theta}}$$

$\downarrow$

directional similarity

$$\cos\theta = \frac{V_1 \cdot V_2}{|V_1|\,|V_2|}$$

measures similarity

Google reports

similar in vector space model

1.

... and 32 more documents like this

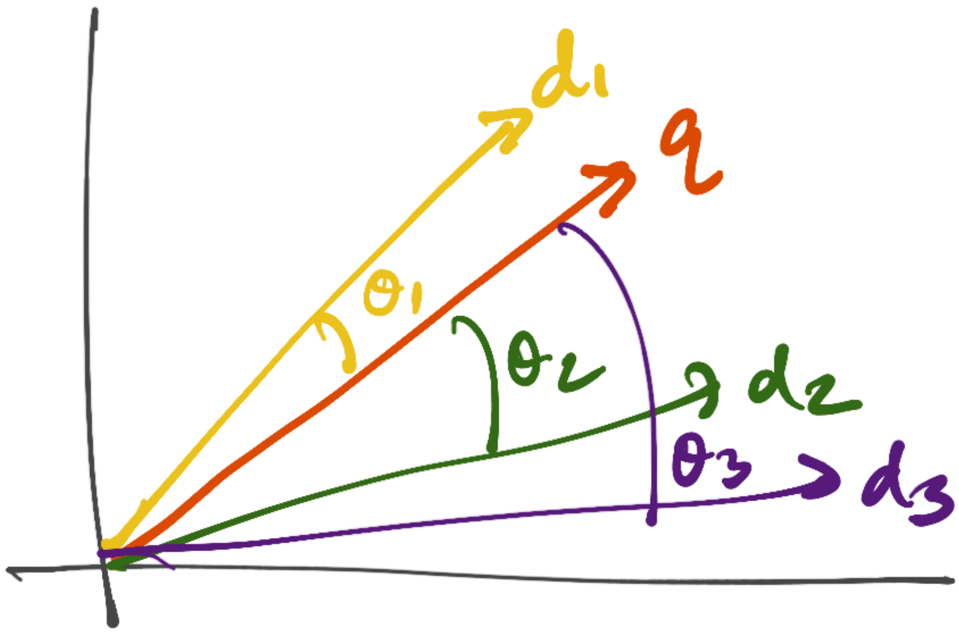
Resolve duplication issue

$$V_2 = 1000 \cdot V_1$$

$$\cos\theta = 1$$

Using vector space model for IR

Treat q also as a vector!



θ_i's give us a ranked response

To compute $\cos\theta$

$$\frac{q \cdot d}{|q| \cdot |d|}$$

Go back to postings and compute the non-normalized version of this

Sufficient to rank θ relative to each other