Classic Information Retrieval III:

Vector model
and
Latent Semantic Indexing

1

Summary weight calculation

- General notation:
  - w_{jd} is the weight of term j in document d
  - $freq_{jd}$ is the # of times term j appears in doc d
  - n_j = # docs containing term j
  - N = number of docs in collection
- Classic *tf-idf* definition of weight:
  $w_{jd} = freq_{jd} * log(N/ n_j)$

2

Weight of query components?

- Set (list) of terms, ***some choices***:
  1. w_{jq} = 0 or 1
  2. $w_{jq} = freq_{jq} * log(N/ n_j)$
     $= 0 \text{ or } log(N/ n_j)$
- Bag of terms
  - Analyze like document
    Some queries are prose expressions of ***information need***

Do we want idf term in both document weights and query weights?

3

Vector Model example

Doc 1: "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies… Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

Frequencies: science 1; knowledge 2; principles 0; engineering 0

Doc 2: "An introduction to computer science in the context of scientific, engineering, and commercial applications. The goal of the course is to teach basic principles and practical issues, while at the same time preparing students to use computers effectively for applications in computer science …" (cos 126 description)

Frequencies: science 2; knowledge 0; principles 1; engineering 1

4

Vector model example cont.

- Consider the 5 100-level and 200-level COS courses as the collection (109, 217, 226)
- Only other appearance of our 4 words is "science" once in 109 description.
- idf: science ln(5/3) = .51
  engineering, principles, knowledge: ln(5/1) = 1.6

5

Term by Doc. Table: $freq_{jd} * log(N/ n_j)$

	Doc 1	Doc 2
science	.51	1.02
engineering		1.6
principles		1.6
knowledge	3.2	

6

Unnormalized score for query:
science, engineering, knowledge, principles
using 0/1 query vector

- Doc 1: 3.71
- Doc 2: 4.22

7

Additional ways to calculate weights

- Dampen frequency effect:
  $w_{jd} = 1 + \log(freq_{jd})$ *if* $freq_{jd} > 0$; *0 otherwise*
- *Use smoothing term to dampen effect:*
  $W_{jd} = a + (1-a)\, freq_{jd} / max_p (freq_{pd})$
  - *a is typically .4 or .5*
  - *Can multiply second term by idf*
- *Effects for long documents (Section 6.4.4)*

8

Where get dictionary of *t* terms?

- Pre-determined dictionary.
  – How sure get all terms?
- Build lexicon when collect documents
  – What if collection dynamic: add docs?

9

Query models advantages

- Boolean
  - No ranking in pure
  + Get power of Boolean Algebra:
    expressivenss and optimize query forms
- Vector
  + Query and document look the same
  + Power of linear algebra
  - No requirement all terms present in pure

10

Latent Semantic Indexing

Developed on board

11