COS 597A:
Principles of
Database and Information Systems

Information Retrieval

Information Retrieval

- Have collection of information "objects":

Text documents	Video
Images	3D models
Audio	…

- User wants information from collection: information need
- User formulates need as a "query"
  - Language of information retrieval system
- System finds objects that "satisfy" query
- System presents objects to user in "useful form"
- User determines which objects from among those presented are relevant

Constrast with Databases

- Relational: structured;
  - defined with schema
- XML: structure + flexibility => semi-structured
  - defined with schema
- Information Retrieval / Search:

  collection of text docs / images / MP3 files …

  may be heterogeneous

  may be many sources with no agreement

  no structure imposed by search system

  No real scheme

What is a query?

relational: SQL query, relational algebra query …

XML: Xpath query, XQuery query, …

General IR: ?

Think first about text documents

- Early digital searches – digital card catalog:
  - subject classifications, keywords
- "Full text" : words + English structure
  - No "meta-structure"
- Classic study
  - Gerald Salton SMART project 1960's
- Lots of scaling since then, but models still helpful

Modeling documents

- *Document* is
  - Set of terms
  - Bag of terms

    duplicates
  - Sequence of terms
- Terms refer to *distinct words or other tokens*
  - numbers, …

Modeling: queries

- *Query*
  - Basic query is one term
  - Multi-term query is
    - List of terms
      - OR model: *some* terms
      - AND model: *all* terms
    - Boolean combination of terms
    - Other constraints?
- Each search engine has own query language
  - similar enough that don't need manual
  - semantics not completely clear

Modeling: "*satisfying*"

- What determines if document satisfies query?
- That depends ….
  - Document model
  - Query model
- *START SIMPLE*
  - *better understanding*
  - *Use components of simple model later*

(pure) Boolean Model of IR

- Document: *set* of terms
- Query: boolean expression over terms
- Satisfying:
  - Doc. evaluates to "true" on single-term query if contains term
  - Evaluate doc. on expression query as you would any Boolean expression
  - doc satisfies query if evals to true on query

Boolean 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)

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)

Query: (principles OR knowledge) AND (science AND NOT(engineering))

Boolean 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)

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)

Query: (principles OR knowledge) AND (science AND NOT(engineering))

Doc 1:	0	1	1	0	TRUE

Boolean 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)

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)

Query: (principles OR knowledge) AND (science AND NOT(engineering))

Doc 2:	1	0	1	1	FALSE

(pure) Boolean Model of IR how "*present results in useful form*"

- most basic: give list
- meaning of order of list? => RANKING?
- There is no ranking algorithm in pure Boolean model
  - Ideas for making one without changing semantics of "satisfy"?

Next Model: Vector Model

- Document: *bag* of terms
- Query: list of terms
- Satisfying:
  - Each document is scored as to the degree it satisfies query (non-negative real number)
  - **doc satisfies query if its score is >0**
  - Documents are returned in sorted list decreasing by score:
    - Include only non-zero scores
    - Include only highest *n* documents, some *n*

How compute score?

1. There is a *dictionary* (aka *lexicon*) of all terms, numbering *t* in all
   - Number the terms 1, …, *t*
2. Represent each document as a *t*-dimensional vector
   - The i^{th} entry of the vector is the *weight* (importance of) term *i* in the document
3. A query is a *t*-dimensional vector
   - The i^{th} entry of the vector is the *weight* (importance of) term *i* in the query

How compute score, continued

4. Calculate a vector function of the document vector and the query vector to get the score of the document with respect to the query.

Choices:

1. Measure the distance between the vectors:

$$\text{Dist}(\boldsymbol{d},\boldsymbol{q}) = \sqrt{(\Sigma^{t}_{i=1}(\boldsymbol{d}_i - \boldsymbol{q}_i)^2)}$$

- Is *dissimilarity* measure
- Not normalized: Dist ranges [0, inf.)
- Fix: use $e^{-\textbf{Dist}}$ with range (0,1]
- Is it the right sense of difference?

How compute score, continued

Choices:

2. Measure the angle between the vectors:

Dot product: $\boldsymbol{d} \cdot \boldsymbol{q} = \Sigma^{t}_{i=1}(\boldsymbol{d}_i * \boldsymbol{q}_i)$

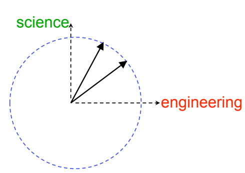
- Is *similarity* measure
- Not normalized: Dist ranges (-inf., inf.)
- Fix: use normalized dot product (cosine), with range [-1,1]

$(\boldsymbol{d} \cdot \boldsymbol{q}) / (|\boldsymbol{d}| * |\boldsymbol{q}|)$ where $|\boldsymbol{v}| = \sqrt{\Sigma^{t}_{i=1}(\boldsymbol{v}_i^2)}$

- In practice vector components are non-negative so range is [0,1]
- This most commonly used function for score

Normalizing vectors

- If use unit vectors, $\boldsymbol{d} / |\boldsymbol{d}|$ and $\boldsymbol{v} / |\boldsymbol{v}|$ some of issues go away



Vector model: Observations

- Have matrix of terms by documents
  ⇒Can use linear algebra
- Queries and documents are the same
  ⇒Can compare documents same way
  - Clustering documents
  - Similarity search
- Document with only some of query terms can score higher than document with all query terms

How compute weights

- Vector model *could* have weights assigned by human intervention
- User setting query weights might make sense
  - User decides importance of terms in own search
- Someone setting document weights makes no sense
  - Huge number documents – billions
- Use model of documents as bag of terms – calculate weights

Some choices for weights

- 0/1 occur/not occur
  - problems?
- term frequency
  - longer docs versus shorter?
    - normalizing helps
  - relative frequency w.r.t other terms?
- weighted term frequence
  - account for frequency of terms in collection
  - can weight for special importance
    - e.g. in title of document - uses some ***structure*** of doc.

Classic 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:

  $w_{jd} = freq_{jd} * \log(N/n_j)$

tf-idf is *"term frequency inverse document frequency"*

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?

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?

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

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

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	

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

- Doc 1: 3.71
- Doc 2: 4.22

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

Other models and variations
probabilistic

Start to enhance model

- Properties of terms within documents?
- Extra information from HTML mark-up?

Properties of terms within documents

- Frequency of term in doc - from Vector model
- Property of each occurrence of term in doc.
  - Where in doc?
  - Special use? (e.g. in title, font, …)
    - HTML tags
- Get general formula for score
  - Weight properties

General Model

- Document: sequence of occurences of terms + attributes
- Query: sequence of terms
  - Can make more complicated
- Docs satisfying query: in current search engines, documents "containing" all terms
  - AND model
- Ranking: wide open function of document and terms

Using Web structure in IR

Hypertext

- document or part of document links to other parts or other documents
  - construct documents of interrelated pieces
  - relate documents to each other
- pre-dates Web
- Web "killer app."

33

How use links to improve information search

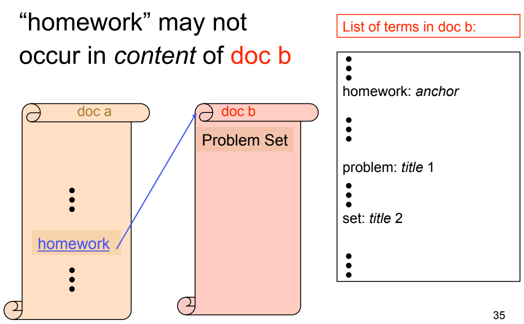
1. use anchor text (HTML)
   - anchor text labels link
   - include anchor text as text of ***document pointed to***
   - may expand vocabulary of document
   - weight?

- Similarly can use words in URL

34

Using anchor text

"homework" may not occur in *content* of doc b



35

General Model

- Document: sequence of occurences of terms + attributes
- Query: sequence of terms
  - Can make more complicated
- Docs satisfying query: in current search engines, documents "containing" all terms
  - AND model
  - "containing" includes anchor text of pointers to this doc from other docs
- Ranking: wide open function of document and terms

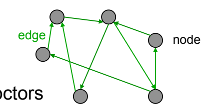
How use links to improve information search?

2. use structure to compute score for ranking

Goal

- Intuition: when Web page points to another Web page, it confers status/authority/popularity to that page
- Find a measure that captures intuition

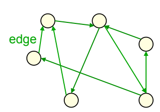
- Not just web linking
  - Citations in books, articles
  - Doctors referring to other doctors



Goal

- Given a directed graph with n nodes
- Assign each node a score that represents its importance in structure
- Most obvious: indegree

  higher indegree => better node
  - Doesn't work well



- We will look at most widely known:

  L. Page and S. Brin's (Google's) PageRank

PageRank

- Algorithm that gave Google the leap in quality
- Used link structure between pages in fundamental way to score pages
  - link structure centerpiece of scoring
- published

  Page, Larry and Sergey Brin, R. Motwani, T. Winograd, *The PageRank Citation Ranking: Bringing Order to the Web,* Stanford Digital Library Technologies Project TR, Jan. 1998.

Conferring importance

Core ideas:

- A node should confer some of its importance to the nodes to which it points
  - If a node is important, the nodes it links to should be important
- A node should not transfer more importance than it has

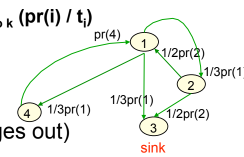
PageRank: Attempt 1

Refer to nodes by numbers 1, … , n (arbitrary numbering)
Let t_i denote the number of edges out of *node i* (outdegree)

Define

$$\mathbf{pr_{new}(k)} = \sum_{\text{i with edge from i to k}} \mathbf{(pr(i) / t_i)}$$

Iterate until converges



Problems
- Sinks (nodes with no edges out)
- Cyclic behavior

PageRank: Attempt 2

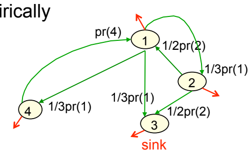
Random walk model

- Attempt 1 gives movement from node to linked neighbor with probability 1/outdegree
- Add random jump to any node

$$\mathbf{pr_{new}(k) = \alpha/n + (1-\alpha)\sum_{i \text{ with edge from i to k}} (pr(i) / t_i)}$$

– α parameter chosen empirically

- Helps break cycles
- Escape from sinks



Normalized?

- Would like $\sum_{1\le k\le n} \mathbf{(pr(k)) = 1}$
- Consider $\sum_{1\le k\le n} \mathbf{(pr_{new}(k))}$

$= \sum_{1\le k\le n}(\alpha/n + (1-\alpha)\sum_{i \text{ with edge from i to k}}(pr(i) / t_i))$

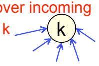
$= \sum_{1\le k\le n}(\alpha/n)+\sum_{1\le k\le n}((1-\alpha)\sum_{i \text{ with edge from i to k}}(pr(i) / t_i))$ *

$= \alpha + (1-\alpha)\sum_{1\le k\le n} \sum_{i \text{ with edge from i to k}}(pr(i) / t_i)$

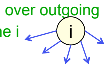
$= \alpha + (1-\alpha)\sum_{1\le i\le n}\sum_{k \text{ with edge from i to k}}(pr(i) / t_i)$ *

$= \alpha + (1-\alpha)\sum_{i \text{ with edge from i}} pr(i))$

*inner sum $\sum_i$ over incoming edges for one k



*inner sum $\sum_k$ over outgoing edges for one i



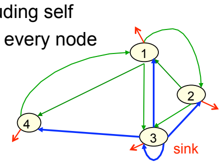
Problem for desired normalization

- Have

$\sum_{1\le k\le n} \mathbf{(pr_{new}(k)) = \alpha + (1-\alpha)\sum_{i \text{ with edge from i}} pr(i))}$

- Missing **pr(i)** for nodes with no edges from them
  – sinks!
- Solution: add n edges out of every sink
  – Edge to every node including self
  – Gives 1/n contribution to every node

Gives desired normalization:
If $\sum_{1\le k\le n} (pr_{initial}(k)) = 1$
then $\sum_{1\le k\le n} (pr(k)) = 1$



Matrix formulation

- Let E be the n by n adjacency matrix
  E(i,k) = 1 if there is an edge from node i to node k
  = 0 otherwise
- Define new matrix L:
  For each row of E ($1\le i\le n$)
  If row i contains t_i >0 ones, $L(i,k)=(1/ t_i) E(i,k)$, $1\le k\le n$
  If row i contains 0 ones, L(i,k) = 1/n for all k
- PageRank defined by equation

$$\boldsymbol{pr} = (\alpha/n, \alpha/n, \dots \alpha/n)^T + (1-\alpha)\, L^T \boldsymbol{pr}$$

has a solution representing the steady-state values pr(k)

Calculation

- Choose α
  – No single best value
  – Page and Brin originally used α=.15
- Simple iterative calculation
  - Initialize $pr_{initial}(k) = 1/n$ for each node k
    - so $\sum_{1\le k\le n} (pr_{initial}(k)) = 1$
  - $pr_{new}(k) = \alpha/n + (1-\alpha)\sum_{1\le i\le n} L(i,k)pr(i)$
- Converges
  – Has necessary mathematical properties
  – In practice, choose convergence criterion
    - Stops iteration

PageRank Observations

- PageRank can be calculated for *any* graph
- Google calculates on entire Web graph
- Huge calculation for Web graph
  – precomputed
  – 1998 Google:
    - 52 iterations for 322 million links
    - 45 iterations for 161 million links
- PageRank must be combined with query-based scoring for final ranking
  – Many variations
  – What Google exactly does secret
  – Can make some guesses by results

Web-based scoring

- PageRank one of class of algorithms
- Second most well-known: HITS
  - designed at same time as PageRank by Jon Kleinberg while at IBM Almaden Research Center
  - Same general goal as PageRank
  - Distinguishes **2 kinds of nodes**
    - **Hubs:** resource pages
      - **Point to many authorities**
    - **Authorities:** good information pages
      - **Pointed to by many hubs**
- **Exploiting Web Structure an important part of information access and analysis**

How use links to improve information search?

3. include more objects to rank

50

Use of HITS

original use **after** find Web pages satisfying query:

1. Retrieve documents satisfy query and rank by term-based techniques
2. Keep top *c* documents: root set of nodes
   – *c* a chosen constant - tunable
3. Make base set:
   a) Root set
   b) *Plus* nodes pointed to by nodes of root set
   c) *Plus* nodes pointing to nodes of root set
4. Make base graph: base set plus edges from Web graph between these nodes
5. Apply HITS to base graph

using links to expand matches!

51

Summary: How use links to improve information search?

- use anchor text (HTML)
  - include anchor text as text of document pointed to
- use structure to compute score for ranking
  - PageRank, HITS
- include more objects to rank
  - saw in use of HITS

❖can deal with objects of mixed types
  – images, PDF, …

52

Social Networks and Ranking

53

Generalized Social Networks

- Represent relationship between entities
  - paper cites paper
  - html page links to html page
  - A supervises B

  directed graph

  - A and B are friends
  - papers share an author
  - A and B are co-workers

  undirected graph

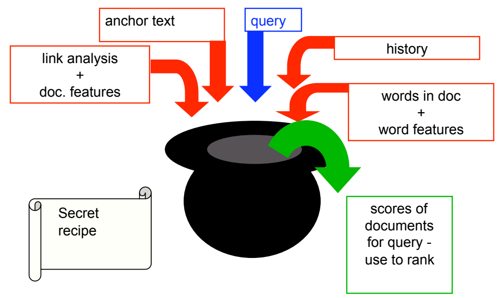
54

History and Personalization

- Modern Web search engines make heavy use of history
  - Aggregate history: what have users done before
  - Personal history: what have you done before
  - Social history: what have friends done

55

Ranking documents w.r.t. query



56

Text Information Retrieval: Summary

- Unstructured or semi-structured data
  - documents
- Each search engine does have precise model of query
- No single correct result
- Ranking determines result

Searching non-text information objects

58

Ways to query for something

1. Query by category/ theme
   - easiest - work done ahead of time
2. Query by describing content
   - text-based query
   - text-based retrieval?
3. Query by example
   - "similar to"
   - imprecise example - sketch

- query text docs and non-text objects with 2
- don't often do doc search by 3
- big move to do music, images by 3

59

Query by example

- How represent objects?
  - features of a class of objects (e.g. image)
  - how compare features?
  - what data structures?
  - what computational methods?
- Issues
  - large number of objects
  - accuracy of representation
  - large size of representation
  - complexity of computations

tradeoffs

tradeoffs

60

Features

- typically vector of numbers characterizing object representation
- "similar to" ≡ close in vector space
  - threshold
  - Euclidean distance?
  - other choices for distance metric

61

Example: content- based image search

62

Example – simple method: color histogram

- k colors
- histogram: % pixels each color
- k×k matrix A of color similarity weights
- histogram defines feature vectors
- $\text{dist}_{histo}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y})^t A(\boldsymbol{x}-\boldsymbol{y})$

$$= \sum_{i=1}^{k}\sum_{j=1}^{k} a_{ij}(x_i-y_i)(x_j-y_j)$$

  - cross-talk: quadratic terms needed
    - not Euclidean distance

63

color histograms: reducing complexity

- compute RED_{avg}, GREEN_{avg}, BLUE_{avg}
  - over all pixels
- use to construct 3D-vector
- use Euclidean distance
- get close candidates
- examine close candidates with full histogram metric

64

color histograms: observations

- works for certain types of images
  - sunset canonical example
- color histogram global property
- this only small part of work:
  QBIC system, IBM, 1995

65

Second example method: a region-based representation

- region-based features of images
- query processed in same way as collection
- space-conscious: use bit vectors
- levels of representation:
  - store bit vector for each region
  - store bit vector for each image
- get close candidates: compare image bit vectors
- compare top k candidates using region bit vectors

66

Image search: Summary of techniques

- Techniques
  - aggregate/average features
  - sample
  - course screening followed by more accurate
- Goals
  - reduce dimension
  - reduce complexity of distance metric
  - reduce space

67

Image search: Commercial search engines

- Use everything you can afford to use
- Text still king!?

68