PURDUE UNIVERSITY | Department of Computer Science

CS47300: Web Information Search and Management

Course Review

Prof. Chris Clifton

4 December 2020

Indiana Center for Database Systems ™

PURDUE UNIVERSITY

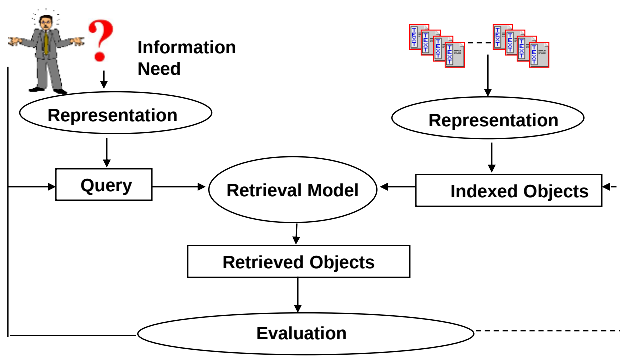
Department of Computer Science

Course Coverage
(Highlights, not exhaustive)

- Text Preprocessing
  - Text representations
  - Stopwords, stemming
- Ad-Hoc IR
  - Crawling, Indexing
  - Retrieval models
    - Boolean
    - TF-IDF
    - Probabilistic: Binary Information Model
    - Latent Semantic Indexing
  - Retrieval Models based on Graph Structure
    - HITS, PageRank
  - Query Expansion, Relevance Feedback
  - Evaluation Metrics
- Text Categorization
  - K-nn, Naïve Bayes, Support Vector Machines, Regression
- Text Clustering
  - K-means
  - Hierarchical
- Collaborative Filtering
  - Memory-based, Model-based
- Natural Language Processing
  - Named Entity Recognition
  - Sentiment Analysis
  - Question Answering
- Map-Reduce
- Ethics Issues
  - Data Privacy
  - Algorithmic Bias/Fairness
- Fake News/Story Detection
- Deep Web & Federated Search

2

AD-hoc IR: Basic Process



Ad-hoc IR: Terminologies

Terminologies:

- Query
  - Representative data of user's information need: text (default) and other media
- Document
  - Data candidate to satisfy user's information need: text (default) and other media
- Database|Collection|Corpus
  - A set of documents
- Corpora
  - A set of databases
  - Valuable corpora from TREC (Text Retrieval Evaluation Conference)

PURDUE UNIVERSITY
Department of Computer Science

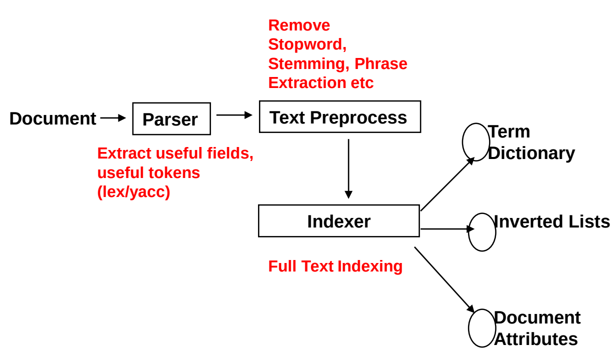
Text Representation

- What to index?
  - All words
    - Stopwords, Stemming
  - Controlled Vocabulary
    - Ontologies
  - Phrases, N-Grams
- How to represent?
  - “Bag of Words” (Vector Space Model)
  - Preserve order, distance

5

PURDUE UNIVERSITY
Department of Computer Science

Text Representation: Process of Indexing



PURDUE UNIVERSITY
Department of Computer Science

Text Representation: Inverted Lists

Doc ID	Text
1	kids question noting in 1960s
2	young man question everything in 1970s
3	kids question questions in 1980s
4	young man question nothing in 2000s

Documents

Term ID	Term	Documents
1	kids	1,3
2	question	1,2,3,4
3	nothing	1,4
4	in	1,2,3,4
5	19060s	1
6	young	2,4
7	man	2,4
8	everything	2
9	1970s	2
10	questions	3
11	1980s	3
11	2000s	4

Inverted Lists

PURDUE UNIVERSITY
Department of Computer Science

Types of Retrieval Models

- Exact Match (Selection) vs. Best Match (Ranking)
- Best Match is usually more accurate/effective
  - Do not need precise query; representative query generates good results
  - Users have control to explore the rank list: view more if need every piece; view less if need one or two most relevant
- Exact Match
  - Hard to define the precise query; too strict (terms are too specific) or too coarse (terms are too general)
  - Users have no control over the returned results
  - Still prevalent in some markets (e.g., legal retrieval)

PURDUE UNIVERSITY
Department of Computer Science

Key Retrieval Models

Retrieval Models

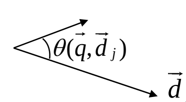
- Boolean — Westlaw
- Vector space
  - Basic vector space/TF-IDF — SMART, LUCENE
  - Extended Boolean
- Probabilistic models
  - Statistical language models — Lemur Project (Indri, Galago)
  - Two Poisson model — Okapi
  - Bayesian inference networks — Inquery
- Citation/Link analysis models
  - Page rank — Google (at one time)
  - Hub & authorities — Clever

15

PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Vector Space Model

- Give two vectors of query and document
- query $\vec{q} = (q_1, q_2, \dots, q_n)$
- document $\vec{d}_j = (d_{j1}, d_{j2}, \dots, d_{jn})$
- calculate the similarity



Cosine similarity: Angle between vectors

$$sim(\vec{q}, \vec{d}_j) = \cos(\theta(\vec{q}, \vec{d}_j))$$

$$\cos\left(\theta\left(\vec{q},\vec{d}_j\right)\right) = \frac{\vec{q}\cdot\vec{d}_j}{\|\vec{q}\|\|\vec{d}_j\|} = \frac{q_1 d_{j,1} + q_1 d_{j,2} + \dots + q_1 d_{j,n}}{\|\vec{q}\|\|\vec{d}_j\|} = \frac{q_1 d_{j,1} + q_1 d_{j,2} + \dots + q_1 d_{j,n}}{\sqrt{q_1^2 + \dots + q_n^2}\sqrt{d_{j,1}^2 + \dots + d_{j,n}^2}}$$

PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Vector Space Model

- Common vector weight components:
- lnc.ltc: widely used term weight
  - "l": log(tf)+1
    - 0 if tf=0
  - "n": no weight/normalization
  - "t": log(N/df)
  - "c": cosine normalization

$$\frac{q_1 d_{j1} + q_2 d_{j2} .. + q_n d_{jn}}{\|\vec{q}\|\|\vec{d}_j\|} = \frac{\sum_k \left[\left(\log(tf_q(k)+1\right)\left(\log(tf_j(k)+1\right)\log\frac{N}{df(k)} \right]}{\sqrt{\sum_k \left[\left(\log(tf_q(k)+1\right)^2\right]}\sqrt{\sum_k \left[\left(\log(tf_j(k)+1\right)\log\frac{N}{df(k)}\right]^2}}$$

21

PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Latent Semantic Indexing

- Latent Semantic Indexing (LSI): Explore correlation between terms and documents
  - Two terms are correlated (may share similar semantic concepts) if they often co-occur
  - Two documents are correlated (share similar topics) if they have many common words
- Associate each term and document with a small number of semantic concepts/topics

22

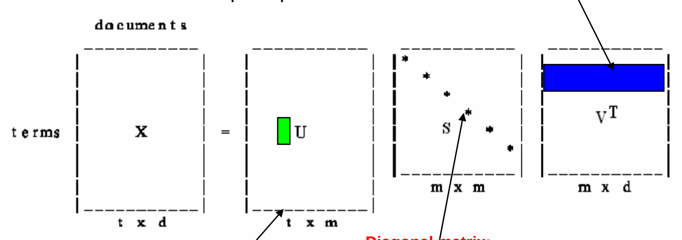
PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Latent Semantic Indexing

- Use singular value decomposition (SVD) to find a small set of concepts/topics

m: number of concepts/topics

Representation of concept in document space; $V^TV=I_m$



Representation of concept in term space; $U^TU=I_m$

Diagonal matrix: concept space

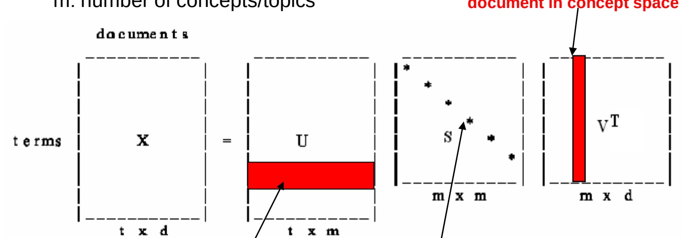
PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Latent Semantic Indexing

- Use singular value decomposition (SVD) to find a small set of concepts/topics

m: number of concepts/topics

Representation of document in concept space



Representation of term in concept space

Diagonal matrix: concept space

PURDUE UNIVERSITY
Department of Computer Science

Retrieval Models: Latent Semantic Indexing

- Retrieval with respect to a query
- Map (fold-in) a query into the representation of the concept space

$$\vec{q}'^T = \vec{q}^T U_k Inv(S_k)$$

- Use the new representation of the query to calculate the similarity between query and all documents
  - Cosine Similarity

27

PURDUE UNIVERSITY
Department of Computer Science

Probability Ranking Principle (PRP)

- Simple case: no selection costs or other utility concerns that would differentially weight errors
- PRP in action: Rank all documents by *p(R=1|x)*
- Theorem: Using the PRP is optimal, in that it minimizes the loss (Bayes risk) under 1/0 loss
  - Provable if all probabilities correct, etc. [e.g., Ripley 1996]
- How do we compute all those probabilities?
  - Do not know exact probabilities, have to use estimates

29

Binary Independence Model

- Queries: binary term incidence vectors
- Given query ***q***,
  - for each document ***d*** need to compute ***p(R|q,d).***
  - replace with computing ***p(R|q,x)*** where ***x*** is the binary term incidence vector representing ***d.***
  - Interested only in ranking
- Use odds and Bayes' Rule:

$$O(R \mid q, \vec{x}) = \frac{p(R=1 \mid q, \vec{x})}{p(R=0 \mid q, \vec{x})} = \frac{\dfrac{p(R=1 \mid q)\, p(\vec{x} \mid R=1, q)}{p(\vec{x} \mid q)}}{\dfrac{p(R=0 \mid q)\, p(\vec{x} \mid R=0, q)}{p(\vec{x} \mid q)}}$$

30

Binary Independence Model

$$O(R \mid q, \vec{x}) = \frac{p(R=1 \mid q, \vec{x})}{p(R=0 \mid q, \vec{x})} = \frac{p(R=1 \mid q)}{p(R=0 \mid q)} \cdot \frac{p(\vec{x} \mid R=1, q)}{p(\vec{x} \mid R=0, q)}$$

Constant for a given query

Needs estimation

- Using **Independence** Assumption:

$$\frac{p(\vec{x} \mid R=1, q)}{p(\vec{x} \mid R=0, q)} = \prod_{i=1}^{n} \frac{p(x_i \mid R=1, q)}{p(x_i \mid R=0, q)}$$

$$O(R \mid q, \vec{x}) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R=1, q)}{p(x_i \mid R=0, q)}$$

31

Binary Independence Model

- Estimating RSV coefficients in theory
- For each term *i* look at this table of document counts:

Documents	Relevant	Non-Relevant	Total
$x_i=1$	s	$n-s$	n
$x_i=0$	$S-s$	$N-n-S+s$	$N-n$
Total	S	$N-S$	N

- Estimates: $p_i \approx \frac{s}{S}$ $r_i \approx \frac{(n-s)}{(N-S)}$

$$c_i \approx K(N,n,S,s) = \log \frac{s/(S-s)}{(n-s)/(N-n-S+s)}$$

For now, assume no zero terms.

35

Estimation – key challenge

- p_i (probability of occurrence in relevant documents) cannot be approximated as easily
- p_i can be estimated in various ways:
  - from relevant documents if know some
    - Relevance weighting can be used in a feedback loop
  - constant (Croft and Harper combination match) – then just get idf weighting of terms (with $p_i=0.5$)

$$RSV = \sum_{x_i=q_i=1} \log \frac{N}{n_i}$$

  - proportional to prob. of occurrence in collection
    - Greiff (SIGIR 1998) argues for 1/3 + 2/3 df_i/N

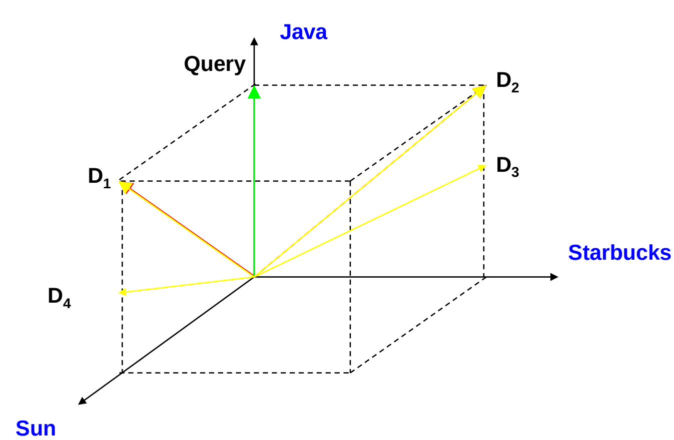
36

Query Expansion

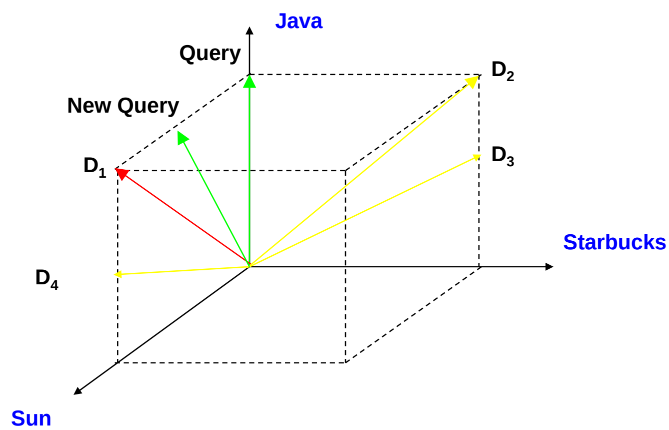
- Users often start with short queries with ambiguous representations
- Observation: Many people refine their queries by analyzing the results from initial queries, or consulting other resources (thesaurus)
  - By adding and removing terms
  - By reweighting terms
  - By adding other features (e.g., Boolean operators)
- Technique of query expansion:

  Can a better query be created automatically?

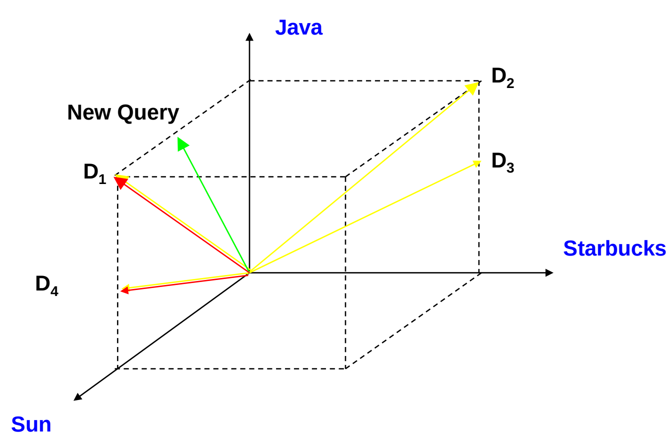
Query Expansion



Query Expansion



Query Expansion



Query Expansion

- Add terms to query to improve recall
  - And possibly precision
- Query Expansion via External Resources
  - Thesaurus
    - "Industrial Chemical Thesaurus", "Medical Subject Headings" (MeSH)
  - Semantic network
    - WordNet
- Relevance Feedback
  - Use user-specified "good documents" to get new terms
  - Blind/Pseudo Relevance Feedback

Blind (Pseudo) Relevance Feedback

- Pseudo-relevance feedback
  - Assume top N (e.g., 20) documents in initial list are relevant
  - Assume bottom N' (e.g., 200-300) in initial list are irrelevant
  - Calculate weights of term according to some criterion (e.g., Rocchio)
  - Select top M (e.g., 10) terms
- Local context analysis
  - Similar approach to pseudo-relevance feedback
  - But use passages instead of documents for initial retrieval; use different term weight selection algorithms

PURDUE UNIVERSITY
Department of Computer Science

Using Link Structure: HITS

- HITS – Hypertext Induced Topic Selection
- For each vertex v ϵ V in a subgraph of interest:
  - a(v) - the authority of v
  - h(v) - the hubness of v
- A site is very authoritative if it receives many citations.
  - Citation from important sites weight more than citations from less-important sites
- Hubness shows the importance of a site.
  - A good hub is a site that links to many authoritative sites

47

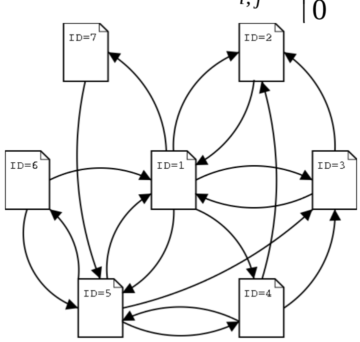
PURDUE UNIVERSITY
Department of Computer Science

Authority and Hub

- Column vector ***a***: a_i is the authority score for the i-th site
- Column vector ***h***: h_i is the hub score for the i-th site

- Matrix **M**:

$$\mathbf{M}_{i,j} = \begin{cases} 1 & \text{the } i\text{th site points to the } j\text{th site} \\ 0 & \text{otherwise} \end{cases}$$



$$\mathbf{M} = \begin{pmatrix} 0&1&1&1&1&0&1 \\ 1&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0 \\ 0&1&1&0&1&0&0 \\ 1&0&1&1&0&1&0 \\ 1&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0 \end{pmatrix}$$

48

Authority and Hub

PURDUE UNIVERSITY
Department of Computer Science

- Column vector ***a***: a_i is the authority score for the i-th site
- Column vector ***h***: h_i is the hub score for the i-th site
- Matrix **M**:

$$\mathbf{M}_{i,j} = \begin{cases} 1 & \text{the } i\text{th site points to the } j\text{th site} \\ 0 & \text{otherwise} \end{cases}$$

Normalization Procedure

- Recursive dependency:

a(v) ← Σ $_{w \in pa[v]}$ h(w)

h(v) ← Σ $_{w \in ch[v]}$ a(w)

$$\mathbf{a}_t = \boxed{\alpha_t} \mathbf{M}^T \mathbf{h}_t$$
$$\mathbf{h}_t = \boxed{\beta_t} \mathbf{M} \mathbf{a}_t$$

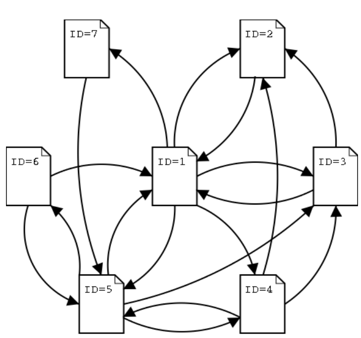
49

Page Rank

PURDUE UNIVERSITY
Department of Computer Science

$$\mathbf{M}_{i,j} = \begin{cases} 1 & \text{the } i\text{th site points to the } j\text{th site} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{B}_{i,j} = \begin{cases} \dfrac{1}{\sum_j \mathbf{M}_{i,j}} & \sum_j \mathbf{M}_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases}$$



$$\mathbf{M} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} 0 & 1/5 & 1/5 & 1/5 & 1/5 & 0 & 1/5 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1/4 & 0 & 1/4 & 1/4 & 0 & 1/4 & 0 \\ 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

50

Matrix Notation

$\mathbf{r} : \mathbf{r}_i$ represents the rank score for the i-th web page

$$r(v) = \alpha \sum_{w \in \mathrm{pa}[v]} \frac{r(w)}{|\mathrm{ch}[w]|'} \longrightarrow$$

$\mathbf{r} = \alpha\, \mathbf{B}^T \mathbf{r}$

α : eigenvalue

r : eigenvector of **B**

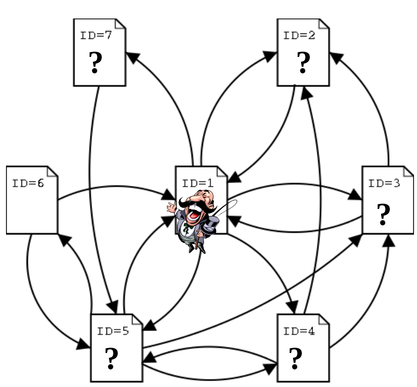
Finding Pagerank

→ find principle eigenvector of B

51

Random Walk Model

- Consider a random walk through the Web graph



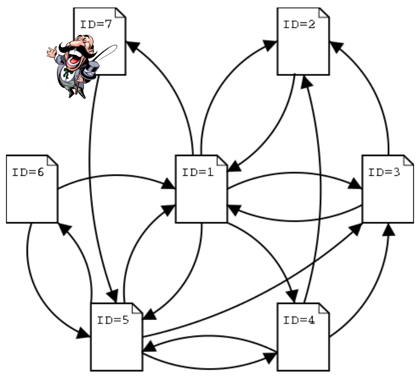
$$\mathbf{B} = \begin{pmatrix} 0 & 1/5 & 1/5 & 1/5 & 1/5 & 0 & 1/5 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 1/4 & 0 & 1/4 & 1/4 & 0 & 1/4 & 0 \\ 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

53

PURDUE UNIVERSITY
Department of Computer Science

Random Walk Model

- Consider a random walk through the Web graph



$$
\mathbf{B} = \begin{pmatrix}
0 & 1/5 & 1/5 & 1/5 & 1/5 & 0 & 1/5 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\
1/4 & 0 & 1/4 & 1/4 & 0 & 1/4 & 0 \\
1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
$$

54

PURDUE UNIVERSITY
Department of Computer Science

Ad-hoc IR Evaluation: Criteria

- Effectiveness
  - Favor returned document ranked lists with more relevant documents at the top
  - Objective measures
    - Recall and Precision
    - Mean-average precision
    - Rank based precision

For documents in a subset of a ranked lists, if we know the truth

	Retrieved	Not retrieved
Relevant	Relevant docs retrieved	Relevant docs not retrieved
Irrelevant	Irrelevant docs retrieved	Irrelevant docs not retrieved

$$\text{Precision} = \frac{\text{Relevant docs retrieved}}{\text{Retrieved docs}}$$

$$\text{Recall} = \frac{\text{Relevant docs retrieved}}{\text{Relevant docs}}$$

60

Evaluation: "Ground Truth"

PURDUE UNIVERSITY
Department of Computer Science

- Manual labeling: Expensive
  – Especially to find "Relevant documents not retrieved"

	Retrieved	Not retrieved
Relevant	Relevant docs retrieved	Relevant docs not retrieved
Irrelevant	Irrelevant docs retrieved	Irrelevant docs not retrieved

- Pooling Strategy
  – Retrieve documents using multiple methods
  – Judge top *n* documents from each method
  – Whole retrieved set is the union of top retrieved documents from all methods
  – Problems: the judged relevant documents may not be complete
  – *It is possible to estimate the total number of relevant documents by random sampling*

63

Evaluation: Single Value Metrics

PURDUE UNIVERSITY
Department of Computer Science

Everyone wants a "score" – which system is best

- Mean average precision
  – Calculate precision at each relevant document; average over all precision values
- 11-point interpolated average precision
  – Calculate precision at standard recall points (e.g., 10%, 20%...); smooth the values; estimate 0 % by interpolation
  – Average the results
- Rank based precision
  – Calculate precision at top ranked documents (e.g., 5, 10, 15…)
  – Desirable when users care more for top ranked documents

64

PURDUE UNIVERSITY
Department of Computer Science

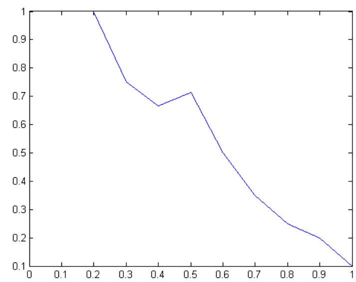
Evaluation Example

- Evaluate a ranked list
  - Precision at Recall
- Evaluate at every relevant document

+
+
-
+
+
-
+

Not Retrieved: +++++

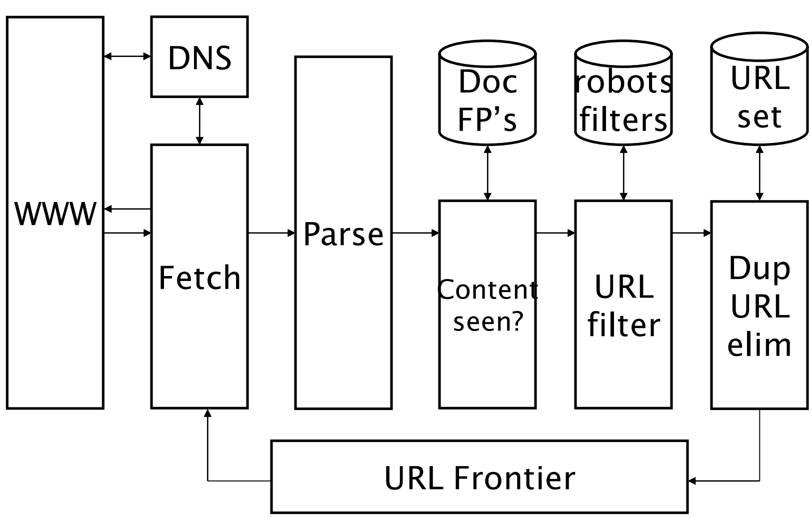
Precision	Recall
1	0.1
1	0.2
0.667	0.2
0.75	0.3
0.8	0.4
0.667	0.4
0.714	0.5



65

PURDUE UNIVERSITY
Department of Computer Science

Basic crawl architecture



68

PURDUE UNIVERSITY
Department of Computer Science

URL frontier: two main considerations

- Politeness: do not hit a web server too frequently
- Freshness: crawl some pages more often than others
  - E.g., pages (such as News sites) whose content changes often

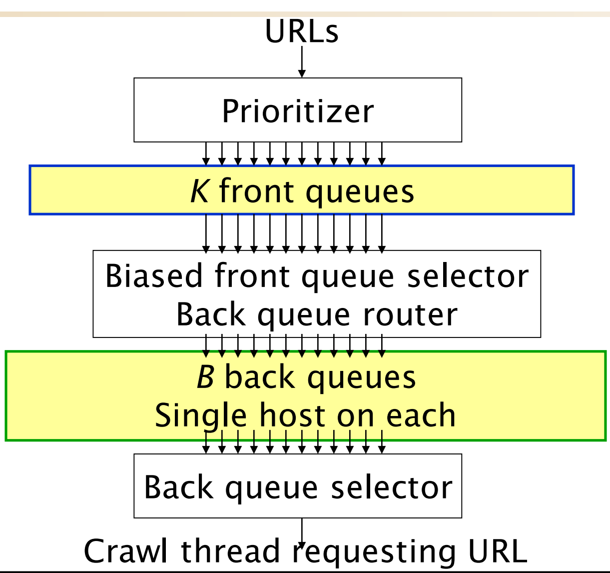
These goals may conflict with each other.

(E.g., simple priority queue fails – many links out of a page go to its own site, creating a burst of accesses to that site.)

69

PURDUE UNIVERSITY
Department of Computer Science

URL frontier: Mercator scheme



70

Purdue University
Department of Computer Science

Age

- Expected age of a page *t* days after it was last crawled:

$$\text{Age}(\lambda, t) = \int_0^t P(\text{page changed at time } x)(t-x)dx$$

- Web page updates follow the Poisson distribution on average
  - time until the next update is governed by an exponential distribution

$$\text{Age}(\lambda, t) = \int_0^t \lambda e^{-\lambda x}(t-x)dx$$

Purdue University
Department of Computer Science

Detecting Duplicates

- Duplicate and near-duplicate documents occur in many situations
  - Copies, versions, plagiarism, spam, mirror sites
  - 30% of the web pages in a large crawl are exact or near duplicates of pages in the other 70%
- Duplicates consume significant resources during crawling, indexing, and search
  - Little value to most users

Computing Similarity

- Features:
  - Segments of a document (natural or artificial breakpoints)
  - Shingles (Word N-Grams)
  - ***a rose is a rose is a rose*** → 4-grams are

    a_rose_is_a

    rose_is_a_rose

    is_a_rose_is

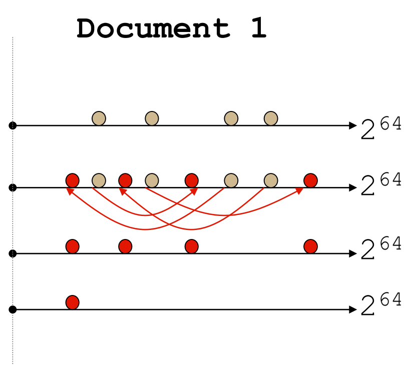
    a_rose_is_a
- Similarity Measure between two docs (= sets of shingles)
  - Jaccard coefficient: (Size_of_Intersection / Size_of_Union)

Sketch of a document

- Create a "sketch vector" (of size ~200) for each document
  - Documents that share ≥ *t* (say 80%) corresponding vector elements are deemed near duplicates
  - For doc *D*, $\text{sketch}_D[\, i\,]$ is as follows:
    - Let f map all shingles in the universe to $1..2^m$ (e.g., f = fingerprinting)
    - Let π_i be a *random permutation* on $1..2^m$
    - Pick MIN $\{\pi_i(f(s))\}$ over all shingles *s* in *D*

Department of Computer Science

Computing Sketch[i] for Doc1



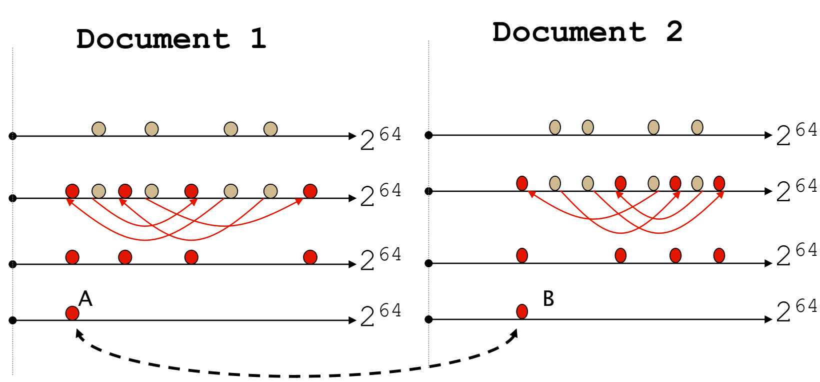
Start with 64-bit *f*(shingles)

Permute on the number line with π_i

Pick the min value

Department of Computer Science

Test if Doc1.Sketch[i] = Doc2.Sketch[i]



Are these equal?

Test for 200 random permutations: π_1, π_2,… π_{200}

Text Categorization

- Task
  - Assign predefined categories to text documents / objects
- Motivation
  - Provide an organizational view of the data
- Procedures
  - Training: Given a set of categories and labeled document examples; learn a method to map a document to correct category (categories)
  - Testing: Predict the category (categories) of a new document

Text Categorization: Evaluation

Contingency Table Per Category (for all docs)

	Truth: True	Truth: False	
Predicted Positive	a	b	a+b
Predicted Negative	c	d	c+d
	a+c	b+d	n=a+b+c+d

a: number of truly positive docs **b: number of false-positive docs**

c: number of false negative docs **d: number of truly-negative docs**

n: total number of test documents

Text Categorization: Evaluation

Recall: r=a/(a+c) **percentage of positive docs detected**

Precision: p=a/(a+b) **how accurate are the predicted positive docs**

Accuracy: (a+d)/n **how accurate are all the predicted docs**

F-measure: $F_\beta = \frac{(\beta^2+1)pr}{\beta^2 p + r}$ $\quad F_1 = \frac{2pr}{p+r}$

Harmonic average: $\frac{1}{\frac{1}{2}\left(\frac{1}{x_1}+\frac{1}{x_2}\right)}$

Error: (b+c)/n **error rate of predicted docs**

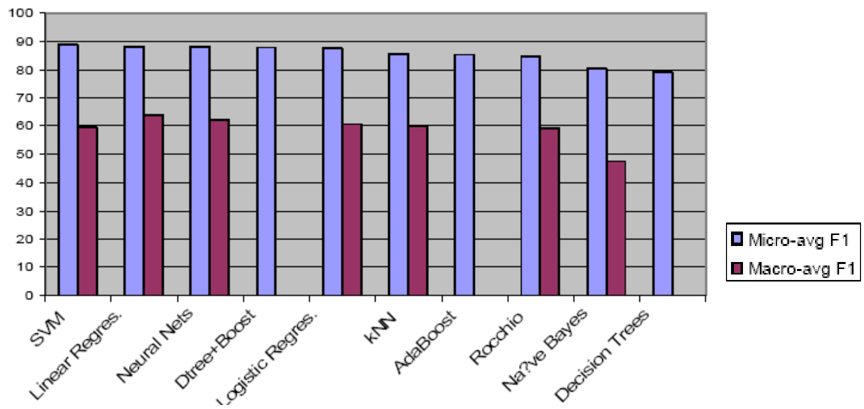
Accuracy+Error=1

Text Categorization: Evaluation

- Micro F1-Measure
  - Calculate a single contingency table for all categories and calculate F1 measure
  - Treat each prediction with equal weight; better for algorithms that work well on large categories
- Macro F1-Measure
  - Calculate a single contingency table for every category; calculate F1 measure separately and average the values
  - Treat each category with equal weight; better for algorithms that work well on many small categories

PURDUE UNIVERSITY
Department of Computer Science

Text Categorization: Evaluation

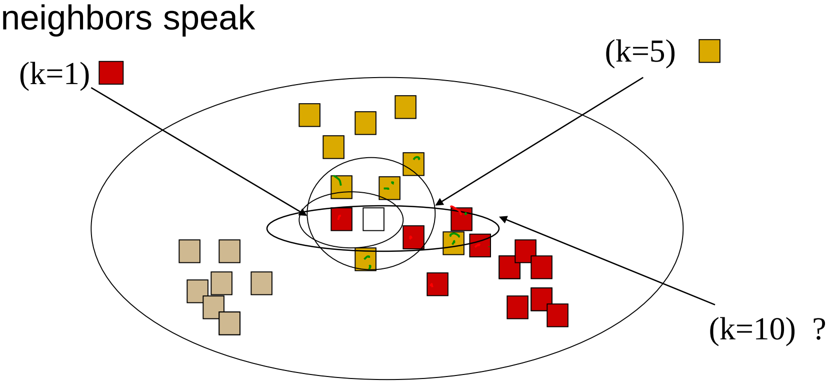


Performance of different algorithms on Reuters-21578 corpus: 90 categories, 7769 Training docs, 3019 test docs, (Yang, JIR 1999)

PURDUE UNIVERSITY
Department of Computer Science

K-Nearest Neighbor Classifier

- Idea: find your language by what language your neighbors speak



- Use K nearest neighbors to vote

1-NN:Red; 5-NN:Brown; 10-NN:?; Weighted 10-NN:Brown

Naïve Bayes Classification

- Naïve Bayes (NB) Classification
  - Generative Model: Model both the input data (i.e., document contents) and output data (i.e., class labels)
  - Make strong assumption of the probabilistic modeling approach
- Methodology
  - Similar with the idea of language modeling approaches for information retrieval
  - Train a language model for all the documents in one category

Naïve Bayes Classification

- Methodology
  - Train a language model for all the documents in one category

$$\text{Category } 1: \left(\vec{d}_{1,1}, \vec{d}_{1,2}, \ldots, \vec{d}_{1,n_1}\right) \rightarrow \text{Language model } \theta_1$$

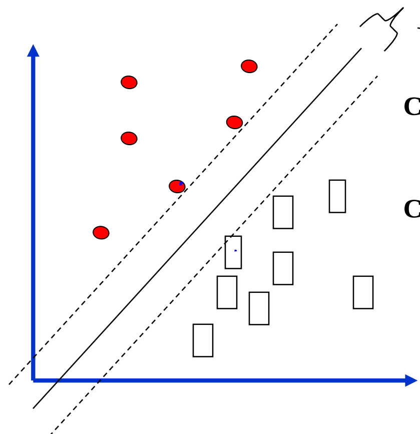
$$\text{Category } 2: \left(\vec{d}_{2,1}, \vec{d}_{2,2}, \ldots, \vec{d}_{2,n_2}\right) \rightarrow \text{Language model } \theta_2$$

......

$$\text{Category C}: \left(\vec{d}_{C,1}, \vec{d}_{C,2}, \ldots, \vec{d}_{C,n_K}\right) \rightarrow \text{Language model } \theta_C$$

  - What is the language model? (Multinomial distribution)
  - How to estimate the language model for all the documents in one category?

Support Vector Machine: *Large-Margin Decision Criterion*



Margin

Closest positive data point to boundary

$$W^T X_i + b = 1$$

Closest negative data point to boundary

$$W^T X_j + b = -1$$

The margin is:

$$m = \frac{2}{||\mathbf{w}||}$$

The Kernel Trick

- Recall the SVM optimization problem

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

Only need inner product

The data points only appear as inner product

As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly

Many common geometric operations (angles, distances) can be expressed by inner products

Define the kernel function K by $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$

Clustering: Issues

- Representation for clustering
  - Document representation
    - Vector space? Normalization?
  - Need a notion of similarity/distance
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid “trivial” clusters - too large or small
      - In an application, if a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.

116

Clustering Algorithms

- Partitioning “flat” algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - *k* means/medoids clustering
    - Model based clustering
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive

117

PURDUE UNIVERSITY
Department of Computer Science

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  – the intra-class (that is, intra-cluster) similarity is high
  – the inter-class similarity is low
  – The measured quality of a clustering depends on both the document representation and the similarity measure used
- External criterion: The quality of a clustering is also measured by its ability to discover some or all of the hidden patterns or latent classes
  – Assessable with gold standard data

123

PURDUE UNIVERSITY
Department of Computer Science

Collaborative Filtering

Objects: O_m

Training Users: U_n

	O_1	O_2	O_3	…… O_j ………	O_M
U_1	3	2	4		
U_2		4	1		1
⋮					
U_i					
⋮					
U_N	5		2		2
Test User U_t	2	3			

$R_u^t(O_j)=$?

What we have:

- Assume there are some ratings by training users
- Test user provides some amount of additional training data

What we do:

- Predict test user's rating based training information

PURDUE UNIVERSITY
Department of Computer Science

Memory-Based Approaches

- How to determine the similarity between users?
  - Measure the similarity in rating patterns between different users

Pearson Correlation Coefficient Similarity

$$w_{u,u^t} = \frac{\sum (R_{u^t}(o) - \bar{R}_{u^t})(R_u(o) - \bar{R}_u)}{\sqrt{\sum (R_{u^t}(o) - \bar{R}_{u^t})^2}\sqrt{\sum (R_u(o) - \bar{R}_u)^2}}$$

Average Ratings

Vector Space Similarity

$$w_{u,u^t} = \frac{\sum R_{u^t}(o) R_u(o)}{\sqrt{\sum R_{u^t}(o)^2}\sqrt{\sum R_u(o)^2}}$$

Prediction: $\longrightarrow$

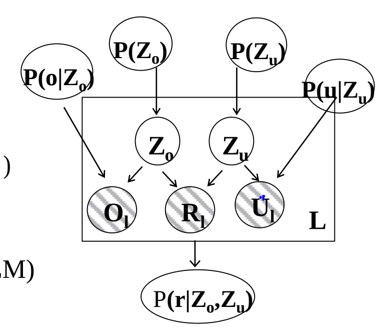
$$\hat{R}_{u^t}(o) = \bar{R}_{u^t} + \frac{\sum_u w_{u,u^t}(R_u(o) - \bar{R}_u)}{\sum_u |w_{u,u^t}|}$$

PURDUE UNIVERSITY
Department of Computer Science

Collaborative Filtering

- Flexible Mixture Model (FMM):

Cluster users and objects separately AND allow them to belong to different classes



$$P(o_{(l)}, u_{(l)}, r_{(l)}) = \sum_{Z_o, Z_u} P(Z_o)P(Z_u)P(o_{(l)} \mid Z_o)P(u_{(l)} \mid Z_u)P(r_{(l)} \mid Z_o, Z_u)$$

- Training Procedure:
  Annealed Expectation Maximization (AEM) algorithm

E-Step: Calculate Posterior Probabilities

$$P(z_o, z_u \mid o_{(l)}, u_{(l)}, r_{(l)}) = \frac{(P(Z_o)P(Z_u)P(o_{(l)} \mid Z_o)P(u_{(l)} \mid Z_u)P(r_{(l)} \mid Z_o, Z_u))^b}{\sum_{Z_o, Z_u}(P(Z_o)P(Z_u)P(o_{(l)} \mid Z_o)P(u_{(l)} \mid Z_u)P(r_{(l)} \mid Z_o, Z_u))^b}$$

PURDUE UNIVERSITY

Department of Computer Science

Collaborative Filtering

$$P(Z_o);\ P(Z_u);\ P(o_{(l)} \mid Z_o);\ P(u_{(l)} \mid Z_u);\ P(r_{(l)} \mid Z_o, Z_u)$$

M-Step: Update Parameters

- Prediction Procedure:
  Fold-In process to calculate joint probabilities

$$P(o, u^t, r_{(l)}) = \sum_{Z_o, Z_u} P(Z_o)P(Z_u)P(o \mid Z_o)P(u^t \mid Z_u)P(r \mid Z_o, Z_u)$$

Fold-in process by EM algorithm

Calculate expectation for prediction

$$\hat{R}_{u^t}(o) = \sum_r r \frac{P(o, u^t, r)}{\sum_{r'} P(o, u^t, r')}$$

"Flexible Mixture Model for Collaborative Filtering", ICML'03

PURDUE UNIVERSITY

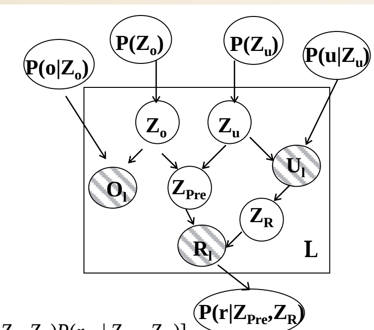
Department of Computer Science

Decoupled Model (DM)

- Decoupled Model (DM):
  Separate preference value

$Z_{pref} \in [1, \ldots, k]$ (1 disfavor, k favor)

from rating $r \in \{1,2,3,4,5\}$



Joint Probability:

$$P(o_{(l)}, u_{(l)}, r_{(l)}) = \sum_{Z_o, Z_u, Z_R} P(Z_o)P(Z_u)P(o_{(l)} \mid Z_o)P(u_{(l)} \mid Z_u)P(Z_R \mid u_{(l)})\Big[\sum_{Z_{pre}} P(Z_{pre} \mid Z_u, Z_o)P(r_{(l)} \mid Z_{pre}, Z_R)\Big]$$

"Preference-Based Graphical Model for Collaborative Filtering", UAI'03

"A study of Mixture Model for Collaborative Filtering", Journal of IR

Content-Based Filtering and Unified Filtering

Content-Based Filtering (CF):

- Generative Methods (e.g. Naïve Bayes)
- Discriminative Methods (e.g. SVM, Logistic Regression)
  - **Usually more accurate**
  - **Can be used to combine features (e.g., actors for movies)**

Unified Filtering by combining CF and CBF:

- Linearly combine the scores from CF and CBF
- Personalized linear combination of the scores
- Bayesian combination with collaborative ensemble learning