COEN281 -- Introduction to Pattern Recognition and Data Mining

Lecture 18: Text Retrieval

Instructor: Dr. Giovanni Seni
GSeni@scu.edu

Department of Computer Engineering
Santa Clara University

Fall/18

Syllabus

Week 1	Introduction; R (Ch.1)
Week 2	Bayesian Decision Theory (Ch.2; DHS: 2.1-2.6, 2.9) Parameter Estimation (DHS: 3.1-3.4)
Week 3	Linear Discriminant Functions (Ch.3&4; DHS: 3.8.2, 5.1-5.8) Regularization (Ch.6; SE: Ch.3)
Week 4	Neural Networks (DHS: 6.1-6.6, 6.8);
Week 5	Support Vector Machines (Ch.9); Introduction to Hadoop & Spark
Week 6	Decision Trees (Ch. 8.1; DHS: 8.3; Ch 2 SE)
Week 7	Ensemble Methods (Ch. 8.2; SE: Ch 4, 5) ;
Week 8	Clustering (Ch. 10; DHS: 10.6, 10.7) Clustering (DHS: 10.9); How many clusters are there? (DHS: 10.10)
Week 9	Non-metric: Association Rules Collaborative Filtering
Week 10	**Text Retrieval**; Other topics

Overview

- Interactive Data Mining
  - Retrieval by Content
- Text Retrieval
  - Vector Space Representation
  - TF-IDF
  - Similarity Measure
  - Latent Semantic Indexing
- Evaluation
  - Precision vs. Recall
- Web Mining - Page Rank

Interactive Data Mining

Retrieval by Content

- Traditional database query
  - Find list of young employees with significant responsibility
  - SQL query:

```
select   employee
from     employees
where    level = manage AND age < 30
```

- Less precise queries
  - Find the k objects in the database that are most similar to:
    - a specific query
    - a specific object
  - E.g., searching the web for reviews of Tai restaurants in San Jose
  - E.g., searching database of satellite images looking for army tanks

Interactive Data Mining

Retrieval by Content (2)

- Problem components
  - Representation
  - Similarity measure
  - Search algorithm
  - How to incorporate user feedback
- Fixed-length vector representation
  - Standard geometric notions of distance
  - Domain independent
  - Significant loss of information
    - E.g., word order, sentence structure

Text Retrieval

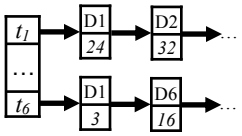
Vector-Space Representation

- Term-Document Matrix
  - The content of a *document* is (approximately) represented by a vector of *term* occurrence counts
    - e.g., $D_i = (w_{i1}, w_{i2}, \ldots w_{iT})^t$ where $w_{ij} = \begin{cases} 1 & doc\text{-}i \text{ contains } term\text{-}j \\ 0 & \text{otherwise} \end{cases}$
  - *document* = {page, article, section,…}
  - *term* = {phrase, word, n-gram,…}
    - Ignore common words: “the”, “and”, etc.
    - Stemming: strip words to root
  - Set of terms t_j defined a priori
    - e.g., t_1 = database, t_2 = SQL, t_3 = index, t_4 = regression, t_5 = likelihood, t_6 = linear

Text Retrieval

Vector-Space Representation (2)

- Term Frequency (TF)
  - $w_{ij} = f_{ij}$ is the number of times that *term-j* appears in *document-i*
  - $w_{ij} = 1+ln(f_{ij})$ sometimes preferred
- Example
  - $N \times T$ sparse matrix
  - T can be very large
  - Inverted file structure



	t_1	t_2	t_3	t_4	t_5	t_6
D1	24	21	9	0	0	3
D2	32	10	5	0	3	0
D3	12	16	5	0	0	0
D4	6	7	2	0	0	0
D5	43	31	20	0	3	0
D6	2	0	0	18	7	16
D7	0	0	1	32	12	0
D8	3	0	0	22	4	2
D9	1	0	0	34	27	25
D10	6	0	0	17	4	23

Text Retrieval

Vector-Space Representation (3)

- Inverse document frequency (IDF)
  - favor terms that occur in relatively few documents
  - $w_{ij} \approx N/n_j$ where n_j is number of documents containing *term-j*
- TF-IDF
  - $w_{ij} = f_{ij} \times ln(N/n_j)$
  - $w_{ij} = (1+ln(f_{ij})) \times (1+ln(N/n_j))$
  - Note how t_1 has been down-weighted

2.53	14.56	4.60	0	0	2.07
3.37	6.93	2.55	0	1.07	0
1.26	11.09	2.55	0	0	0
0.63	4.85	1.02	0	0	0
4.53	21.48	10.21	0	1.07	0
0.21	0	0	12.47	2.50	11.09
0	0	0.51	22.18	4.28	0
0.31	0	0	15.24	1.42	1.38
0.10	0	0	23.56	9.63	17.33
0.63	0	0	11.78	1.42	15.94

Text Retrieval

Document Similarity

- Cosine Measure

$$D_i^t \bullet D_j \quad \text{or} \quad \frac{D_i^t \bullet D_j}{\|D_i\| \|D_j\|}$$

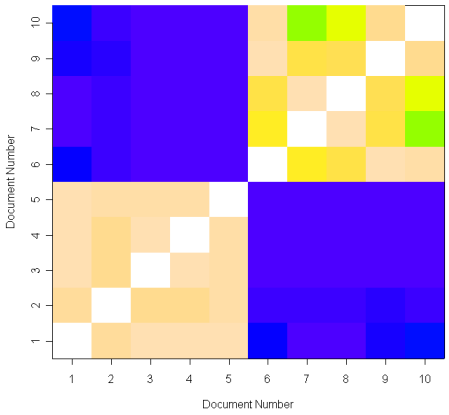
$$= \frac{\sum_{k=1}^{T} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{T} w_{ik}^2 \sum_{k=1}^{T} w_{jk}^2}}$$

- Queries are also (short) documents
  - $Q = (q_1, q_2, \ldots q_T)^t$
  - e.g., Find: linear regression
    $\Rightarrow$ $Q = (0, 0, 0, 1, 0, 1)^t$

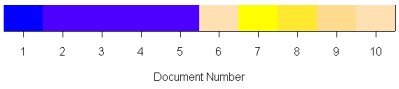
Text Retrieval

Document Similarity (2)

- Pairwise document similarity



- Document-query similarity



Text Retrieval

Latent Semantic Indexing

- Do "data mining" queries find "knowledge discovery" documents?
  - Expand the query with list of "synonyms"
- How to link *semantically* related terms?
  - Thesaurus ontology
  - Data driven approach
    - Fit a SVD to the $N \times T$ document-term matrix
    - Approximate original T–dimensional space by the first k principal component directions
      - A singular vector might contain non-zero coefficients on semantically related terms
      - Easier to store: $N \times k$ with little loss in information

Text Retrieval

Latent Semantic Indexing (2)

- The SVD of a $N \times T$ matrix X has the form

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^t \quad \text{where } \mathbf{U} \text{ is } N \times T \text{ orthogonal (ie. } \mathbf{U}^t\mathbf{U} = \mathbf{I})$$
$$\mathbf{V} \text{ is } T \times T \text{ orthogonal}$$
$$\mathbf{D} \text{ is } T \times T \text{ diagonal}$$

(hanger)(stretcher)(aligner)

- $\mathbf{V^t}$ - principal component directions
  - Best set of orthogonal basis vectors in terms of explaining the most variance in the data
- $\mathbf{D}$ - with diagonal entries $d_1 \geq d_2 \geq \ldots \geq d_T \geq 0$ called singular values of X
- $\mathbf{U}$ - data representation in principal component space

Text Retrieval

Latent Semantic Indexing (3)

- For our *10×6* matrix:
  - Singular values: *{45.9, 32.1, 15.8, 4.6, 2.8, 1.9}*
  - If we retain first two principal components: $(d_1^2 + d_2^2) / \sum_{i=1}^{6} d_i^2 = 0.9169$
    - i.e., only *8.3%* of the information has been lost
  - Represent documents in new 2-dimmensional space:

```
> a.svd <- svd(a)
> a.svd$u[,1:2]
```

	[,1]	[,2]
[1,]	-0.031282706	0.479678180
[2,]	-0.010483106	0.245127457
[3,]	-0.006716370	0.350324638
[4,]	-0.002935571	0.152603387
[5,]	-0.020396412	0.749887135
[6,]	-0.358808158	-0.010914926
[7,]	-0.429621555	-0.023447358
[8,]	-0.302415920	-0.018738738
[9,]	-0.663254484	-0.020356602
[10,]	-0.392124830	-0.006408339

Text Retrieval

Latent Semantic Indexing (4)

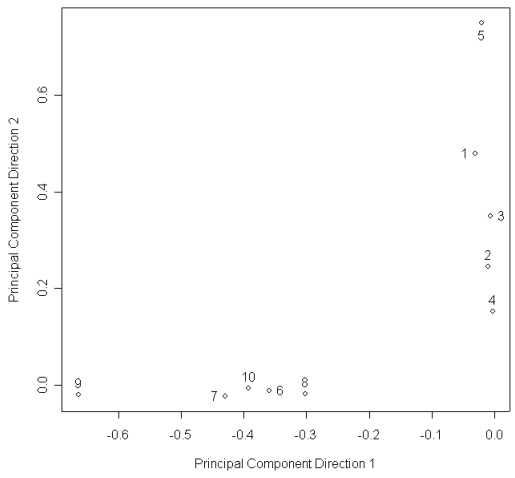
- Projected location of the 10 documents
  - pseudo-terms:

```
> a.svd$v[,1:2]
```

	[,1]	[,2]
[1,]	-0.01523895	0.18512280
[2,]	-0.02297811	0.91367130
[3,]	-0.01346301	0.35797380
[4,]	-0.84635909	-0.04649298
[5,]	-0.22090797	0.02190653
[6,]	-0.48366774	0.01214782

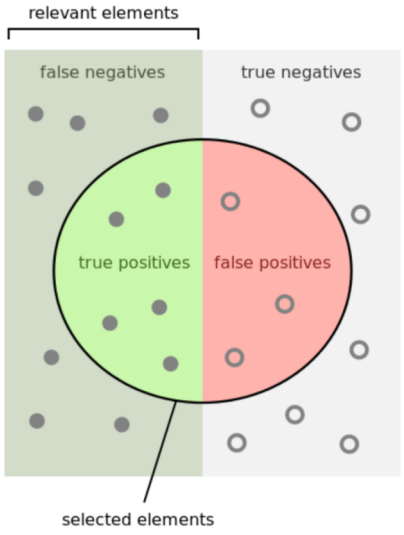
  - To convert a query into pseudo-term representation:

$$\tilde{Q}_{1\times 2} = Q_{1\times 6} \times [v_1 \quad v_2]_{6\times 2}$$



Text Retrieval Evaluation

Precision vs. Recall



- How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

- How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

Image source: Wikipedia

Web Mining

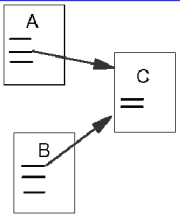
PageRank

- Web is not a "flat" collection of text documents
  - Hyperlink structure provides auxiliary information to rank matches
- Augmented text search
  - Find pages matching query
  - Rank matching pages based on "quality"
- Quality analogy in standard citation analysis
  - More citations ⇒ higher importance
- What is a web page citation? backlinks!

Web Mining

PageRank (2)

- Web as a directed graph
  - 4,285,199,774 nodes (pages)
  - 1 trillion ? edges (links)



- Is simple citation count enough?
  - A link from Yahoo should be weighted more than a link from an obscure site
- PageRank: a page has high rank if the sum of its backlinks is high
  - Rank $R(u) = c\sum_{v \in B_u} \frac{R(v)}{N_v}$

B_u : set of pages that point to u

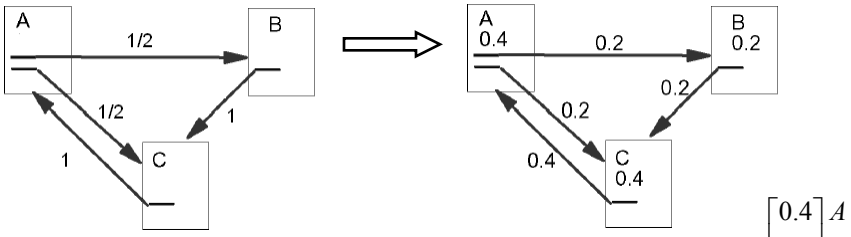
F_u : set of pages u points to

$N_u = |F_u|$

Web Mining

PageRank (3)

- PageRank equation is recursive
  - Idealized calculation
    - Start with any rank values
    - Iterate until convergence – i.e., R' s no longer changes



steady state $R = \begin{bmatrix} 0.4 \\ 0.2 \\ 0.4 \end{bmatrix} \begin{matrix} A \\ B \\ C \end{matrix}$

Web Mining

PageRank (4)

- PageRank as dominant eigenvector
  - Consider "adjacency matrix" representation of graph (row/column normalized)

$$M[u,v] = \begin{cases} 1/N_u & \text{there is an edge } u \to v \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{bmatrix} 0 & 0 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 1 & 0 \end{bmatrix} \end{array}$$

  - Eigenvectors of M: solution to $\boldsymbol{Mx} = \lambda \boldsymbol{x}$
  - Markov chain theory results
    - A stationary state satisfies $\pi = M\pi$, $\Sigma\pi_i = 1$, $\pi_i \geq 0$
    - Largest (dominant) eigenvalue is always 1 (i.e., λ=1)
  - Power method of computing eigenvectors $\boldsymbol{\pi}^{(k)} = \mathbf{M}\mathbf{R}^{(k-1)}$; $\mathbf{R}^{(k)} = \dfrac{\boldsymbol{\pi}^{(k)}}{\left\|\boldsymbol{\pi}^{(k)}\right\|}$

Web Mining

PageRank (5)

- Why it works well?
  - Great for underspecified queries (e.g., query="Stanford University")
  - Relatively immune to commercial manipulation
    - For a page to get a high PageRank it must convince an important page, or many non-important pages to link to it
  - Trustworthy: link vote > click vote
    - Only requires public links
- Problems?
  - "rich get richer" syndrome, seldom returns yahoo-like pages (hubs),..
  - See http://www.google-watch-watch.org/pagerank.php