Text Analytics (Text Mining)

Concepts, Algorithms, LSI/SVD

Duen Horng (Polo) Chau
Assistant Professor
Associate Director, MS Analytics
Georgia Tech

Partly based on materials by
Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos, Parishit Ram (GT PhD alum; SkyTree), Alex Gray
Text is everywhere

We use documents as primary information artifact in our lives

Our access to documents has grown tremendously thanks to the Internet

- **WWW**: webpages, Twitter, Facebook, Wikipedia, Blogs, ...
- **Digital libraries**: Google books, ACM, IEEE, ...
- Lyrics, closed caption... (youtube)
- Police case reports
- Legislation (law)
- Reviews (products, rotten tomatoes)
- Medical reports (EHR - electronic health records)
- Job descriptions
Big (Research) Questions

... in understanding and gathering information from text and document collections

- establish authorship, authenticity; plagiarism detection
- classification of genres for narratives (e.g., books, articles)
- tone classification; sentiment analysis (online reviews, twitter, social media)
- code: syntax analysis (e.g., find common bugs from students’ answers)
Popular **Natural Language Processing (NLP) libraries**

- **Stanford NLP**
  - tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing

- **OpenNLP**

- **NLTK (python)**

---

**Named Entity Recognition:**

```
1. President Xi Jinping of China, on his first state visit to the United States, showed off his familiarity with American history and pop culture on Tuesday night.
```

**Coreference:**

```
1. President Xi Jinping of China, on his first state visit to the United States, showed off his familiarity with American history and pop culture on Tuesday night.
```

**Basic Dependencies:**

---

Image source: https://stanfordnlp.github.io/CoreNLP/
Outline

• **Preprocessing** (e.g., stemming, remove stop words)

• **Document representation** (most common: bag-of-words model)

• **Word importance** (e.g., word count, TF-IDF)

• **Latent Semantic Indexing** (find “concepts” among documents and words), which helps with retrieval

To learn more: Prof. Jacob Eisenstein’s CS 4650/7650 Natural Language Processing
Stemming

Reduce words to their **stems** (or base forms)

**Words:** compute, computing, computer, ...

**Stem:** comput

Several classes of algorithms to do this:

- Stripping suffixes, lookup-based, etc.


Bag-of-words model

Represent each document as a bag of words, ignoring words’ ordering. Why? For simplicity.

Unstructured text becomes a vector of numbers e.g., docs: “I like visualization”, “I like data”.

1 : “I”
2 : “like”
3 : “data”
4 : “visualization”

“I like visualization” ➔ [1, 1, 0, 1]
“I like data” ➔ [1, 1, 1, 0]
TF-IDF

A word’s importance score in a document, among N documents

When to use it? Everywhere you use “word count”, you can likely use TF-IDF.

TF: term frequency
= #appearance a document
(high, if terms appear many times in this document)

IDF: inverse document frequency
= log( N / #document containing that term)
(penalize “common” words appearing in almost any documents)

Final score = TF * IDF
(higher score ➞ more “characteristic”)

Example: http://en.wikipedia.org/wiki/Tf%E2%80%93idf#Example_of_tf%E2%80%93idf
Vector Space Model

Why?

Each document ➞ vector
Each query ➞ vector

Search for documents ➞ find “similar” vectors
Cluster documents ➞ cluster “similar” vectors
Latent Semantic Indexing (LSI)

Main idea

- map each document into some ‘concepts’
- map each term into some ‘concepts’

‘Concept’ : ~ a set of terms, with weights.

For example, DBMS_concept:
  “data” (0.8),
  “system” (0.5),
Latent Semantic Indexing (LSI)
~ pictorially \((\text{before})\) ~

document-term matrix

<table>
<thead>
<tr>
<th></th>
<th>data</th>
<th>system</th>
<th>retrieval</th>
<th>lung</th>
<th>ear</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>doc2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>doc3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>doc4</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Latent Semantic Indexing (LSI) ~ pictorially (after) ~

<table>
<thead>
<tr>
<th>database concept</th>
<th>medical concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>1</td>
</tr>
<tr>
<td>system</td>
<td>1</td>
</tr>
<tr>
<td>retrieval</td>
<td>1</td>
</tr>
<tr>
<td>lung</td>
<td>1</td>
</tr>
<tr>
<td>ear</td>
<td>1</td>
</tr>
</tbody>
</table>

... and

<table>
<thead>
<tr>
<th>database concept</th>
<th>medical concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc1</td>
<td>1</td>
</tr>
<tr>
<td>doc2</td>
<td>1</td>
</tr>
<tr>
<td>doc3</td>
<td>1</td>
</tr>
<tr>
<td>doc4</td>
<td>1</td>
</tr>
</tbody>
</table>
Q: How to search, e.g., for “system”?
A: find the corresponding concept(s); and the corresponding documents

Latent Semantic Indexing (LSI)

<table>
<thead>
<tr>
<th>database concept</th>
<th>medical concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>1</td>
</tr>
<tr>
<td>system</td>
<td>1</td>
</tr>
<tr>
<td>retrieval</td>
<td>1</td>
</tr>
<tr>
<td>lung</td>
<td>1</td>
</tr>
<tr>
<td>ear</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>database concept</th>
<th>medical concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc1</td>
<td>1</td>
</tr>
<tr>
<td>doc2</td>
<td>1</td>
</tr>
<tr>
<td>doc3</td>
<td>1</td>
</tr>
<tr>
<td>doc4</td>
<td>1</td>
</tr>
</tbody>
</table>
Latent Semantic Indexing (LSI)

Works like an automatically constructed thesaurus

We may retrieve documents that DON’T have the term “system”, but they contain almost everything else (“data”, “retrieval”)
LSI - Discussion

Great idea,
• to derive ‘concepts’ from documents
• to build a ‘thesaurus’ automatically
• to reduce dimensionality (down to few “concepts”)

How does LSI work?
Uses **Singular Value Decomposition** (SVD)
Singular Value Decomposition (SVD)
Motivation

Problem #1
Find “concepts” in matrices

Problem #2
Compression / dimensionality reduction

<table>
<thead>
<tr>
<th></th>
<th>bread</th>
<th>lettuce</th>
<th>tomatoes</th>
<th>beef</th>
<th>chicken</th>
</tr>
</thead>
<tbody>
<tr>
<td>vegetarians</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
SVD is a powerful, generalizable technique.

<table>
<thead>
<tr>
<th>Customers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Songs / Movies / Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
SVD Definition (pictorially)

\[ A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} (V_{[m \times r]})^T \]

- **Diagonal matrix**
  - Diagonal entries: concept strengths

- **m terms**
- **r concepts**
SVD Definition (in words)

\[ A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} (V_{[m \times r]})^T \]

**A**: \( n \times m \) matrix  
  e.g., \( n \) documents, \( m \) terms

**U**: \( n \times r \) matrix  
  e.g., \( n \) documents, \( r \) concepts

**\( \Lambda \)**: \( r \times r \) diagonal matrix  
  \( r \): rank of the matrix; strength of each ‘concept’

**V**: \( m \times r \) matrix  
  e.g., \( m \) terms, \( r \) concepts
**SVD - Properties**

**THEOREM** [Press+92]:

always possible to decompose matrix $A$ into

$$A = U \Lambda V^T$$

$U$, $\Lambda$, $V$: unique, most of the time

$U$, $V$: column **orthonormal**

i.e., columns are **unit vectors**, and **orthogonal** to each other

$$U^T U = I$$

$$V^T V = I$$

($I$: identity matrix)

$\Lambda$: **diagonal** matrix with non-negative diagonal entries, sorted in **decreasing order**
SVD - Example

\[
A = UV^T
\]

\[
\begin{pmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0.18 & 0 \\
0.53 & 0 \\
0.80 & 0 \\
0.27 & 0 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
9.64 & 0 \\
0 & 5.29 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \ \\
0 & 0 & 0 & 0.71 & 0.71 \ \\
\end{pmatrix}
\]
SVD - Example

CS docs = \begin{pmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{pmatrix}

MD docs = \begin{pmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
9.64 & 0 \\
0.53 & 0 \\
0.80 & 0 \\
0.27 & 0 \\
\end{pmatrix}

“strength” of CS-concept

CS concept x MD concept = \begin{pmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71 \\
\end{pmatrix}

document-concept similarity matrix

term-concept similarity matrix
SVD - Interpretation #1

‘documents’, ‘terms’ and ‘concepts’:

\( \mathbf{U} \): document-concept similarity matrix
\( \mathbf{V} \): term-concept similarity matrix
\( \Lambda \): diagonal elements: concept “strengths”
SVD - Interpretation #1

‘documents’, ‘terms’ and ‘concepts’:
Q: if $A$ is the document-to-term matrix, what is the similarity matrix $A^T A$?
A:

Q: $A A^T$?
A:
SVD - Interpretation #1

‘documents’, ‘terms’ and ‘concepts’:

Q: if $A$ is the document-to-term matrix, what is the similarity matrix $A^T A$?
A: term-to-term ($[m \times m]$) similarity matrix

Q: $A A^T$?
A: document-to-document ($[n \times n]$) similarity matrix
SVD properties

V are the eigenvectors of the covariance matrix $A^TA$

$$A^TA = (U\Sigma V^T)^T(U\Sigma V^T) = V\Sigma^2V^T$$

U are the eigenvectors of the Gram (inner-product) matrix $AA^T$

$$AA^T = (U\Sigma V^T)(U\Sigma V^T)^T = U\Sigma^2U^T$$

SVD is closely related to PCA, and can be numerically more stable.

For more info, see:


SVD - Interpretation #2

Find the best axis to project on.

('best' = min sum of squares of projection errors)

Beautiful visualization explaining PCA:
http://setosa.io/ev/principal-component-analysis/
**SVD - Interpretation #2**

\[ A = U \Lambda V^T \]

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

\[ \begin{bmatrix}
0.18 & 0 & 0.36 & 0 & 0.18 & 0 & 0.90 & 0 & 0.18 & 0 & 0.90 & 0 & 0.53 & 0 & 0.80 & 0 & 0.27 & 0
\end{bmatrix} \times \begin{bmatrix}
9.64 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \times \begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 & 0.71 & 0.71
\end{bmatrix} \]

- Variance (‘spread’) on the v1 axis.
- First Singular Vector.
$U \Lambda$ gives the coordinates of the points in the projection axis

$\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix} = \begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix} \times \begin{bmatrix}
9.64 & 0 \\
0 & 5.29 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix} \times \begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}$
SVD - Interpretation #2

More details
Q: how exactly is dim. reduction done?
**SVD - Interpretation #2**

More details

Q: how exactly is dim. reduction done?
A: set the smallest singular values to zero:

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
= \begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
9.64 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\times
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
\]
SVD - Interpretation #2

More details

Q: how exactly is dim. reduction done?
A: set the smallest singular values to zero:

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
= \begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27 \\
\end{bmatrix}
\times \begin{bmatrix}
9.64 & 0 \\
0 & 9.64 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
\end{bmatrix}
\times \begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0.71 & 0.71 & 0 & 0 \\
\end{bmatrix}
\]
SVD - Interpretation #2

More details

Q: how exactly is dim. reduction done?
A: set the smallest singular values to zero:

\[
\begin{pmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{pmatrix}
= \begin{pmatrix}
0.18 \\
0.36 \\
0.18 \\
0.90 \\
\end{pmatrix}
\times
\begin{pmatrix}
9.64 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
\end{pmatrix}
\times
\begin{pmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
\end{pmatrix}
\]
SVD - Interpretation #2

More details

Q: how exactly is dim. reduction done?
A: set the smallest singular values to zero:
SVD - Interpretation #3

finds non-zero ‘blobs’ in a data matrix

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27 \\
\end{bmatrix}
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29 \\
\end{bmatrix}
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71 \\
\end{bmatrix}
\]
SVD - Interpretation #3

finds non-zero ‘blobs’ in a data matrix

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
= \begin{bmatrix}
0.18 & 0 & \\
0.36 & 0 & \\
0.18 & 0 & \\
0.90 & 0 & \\
0 & 0.53 & \\
0 & 0.80 & \\
0 & 0.27 & \\
\end{bmatrix}
\begin{bmatrix}
9.64 & 0 & \\
0 & 5.29 & \\
\end{bmatrix}
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71 \\
\end{bmatrix}
\]
### SVD - Interpretation #3

- finds non-zero ‘blobs’ in a data matrix =
- ‘communities’ (bi-partite cores, here)

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\]

Row 1  Row 4  Row 5  Row 7

Col 1  Col 3  Col 4
SVD - Complexity

$O(n \times m \times m)$ or $O(n \times n \times m)$ (whichever is less)

Faster version, if just want singular values
or if we want first $k$ singular vectors
or if the matrix is sparse [Berry]

No need to write your own!
Available in most linear algebra packages
(LINPACK, matlab, Splus/R, mathematica ...)
Case Study
How to do queries with LSI?
Case Study

How to do queries with LSI?

For example, how to find documents with ‘data’?

```
<table>
<thead>
<tr>
<th></th>
<th>data</th>
<th>info</th>
<th>retrieval</th>
<th>brain</th>
<th>lung</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS docs</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MD docs</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th></th>
<th>0.18</th>
<th>0</th>
<th>0.36</th>
<th>0</th>
<th>0.18</th>
<th>0</th>
<th>0.90</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>9.64</td>
<td>0</td>
<td>0</td>
<td>5.29</td>
<td>0</td>
<td>0.53</td>
<td>0</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>5.29</td>
<td>0</td>
<td>0</td>
<td>0.53</td>
<td>0</td>
<td>0.71</td>
<td>0.71</td>
</tr>
</tbody>
</table>
```
Case Study

How to do queries with LSI?

For example, how to find documents with ‘data’?
A: map query vectors into ‘concept space’ – how?

\[
\text{CS docs} \uparrow \\
\begin{array}{c|c|c|c|c}
\text{data} & \text{info} & \text{retrieval} & \text{brain} & \text{lung} \\
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1 \\
\end{array}
\]

\[
\text{MD docs} \downarrow \\
\begin{array}{c}
0.18 \\
0.36 \\
0.18 \\
0.90 \\
9.64 \\
0.53 \\
0.80 \\
0.27 \\
\end{array}
\]

\[
= \times \times
\]

\[
\begin{array}{c|c|c|c|c}
\text{docs} & \text{data} & \text{info} & \text{retrieval} & \text{brain} \\
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71 \\
\end{array}
\]
Case Study

How to do queries with LSI?

For example, how to find documents with ‘data’?

A: map query vectors into ‘concept space’, using **inner product** (cosine similarity) with each ‘concept’ vector \( v_i \)

\[
\begin{bmatrix}
data & info & retrieval & brain & lung \\
1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
q = v_1 \circ q
\]

\[
v_2
\]
Case Study

How to do queries with LSI?

Compactly, we have:

$$q V = q_{\text{concept}}$$

**Example:**

```
<table>
<thead>
<tr>
<th>data</th>
<th>info</th>
<th>retrieval</th>
<th>brain</th>
<th>lung</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

**Term-concept similarity matrix**

```
<table>
<thead>
<tr>
<th></th>
<th>data</th>
<th>info</th>
<th>retrieval</th>
<th>brain</th>
<th>lung</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>0.58</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>info</td>
<td>0.58</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>retrieval</td>
<td>0.58</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>brain</td>
<td>0</td>
<td>0.71</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lung</td>
<td>0</td>
<td>0</td>
<td>0.71</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

CS concept
Case Study

How would the document (‘information’, ‘retrieval’) be handled?
Case Study

How would the document ('information', 'retrieval') be handled?

$$d \, V = d_{\text{concept}}$$

### Term-Concept Similarity Matrix

<table>
<thead>
<tr>
<th></th>
<th>data</th>
<th>info</th>
<th>retrieval</th>
<th>brain</th>
<th>lung</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>info</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>retrieval</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>brain</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lung</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Concept Similarity (CS) Matrix

<table>
<thead>
<tr>
<th></th>
<th>concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>1.16</td>
</tr>
<tr>
<td>info</td>
<td>0</td>
</tr>
<tr>
<td>retrieval</td>
<td>0</td>
</tr>
<tr>
<td>brain</td>
<td>0</td>
</tr>
<tr>
<td>lung</td>
<td>0</td>
</tr>
</tbody>
</table>

**SAME!**
Document ('information', 'retrieval') will be retrieved by query ('data'), even though it does not contain 'data'!!
Case study - LSI

Q1: How to do queries with LSI?

Q2: multi-lingual IR (english query, on spanish text?)
Case study - LSI

• Problem:
  – given many documents, translated to both languages (e.g., English and Spanish)
  – answer queries across languages
Case study - LSI

• Solution: ~ LSI
Switch Gear to Text Visualization
Word/Tag Cloud (still popular?)

http://www.wordle.net
http://www.jasondavies.com/wordtree/
Phrase Net

Visualize pairs of words satisfying a pattern (“X and Y”)
Termite: Topic Model Visualization

http://vis.stanford.edu/papers/termite
Termite: Topic Model Visualization

http://vis.stanford.edu/papers/termite

Using “Seriation”