Matrix Decomposition and Latent Semantic Indexing (LSI)

Introduction to Information Retrieval
CS 221
Donald J. Patterson
Outline

• Introduction

• Linear Algebra Refresher
Star Cluster NGC 290 - ESA & NASA

- A picture of the sky is two dimensional
- The stars are not in two dimensions
- When we take a photo of stars we are projecting them into 2-D
  - projecting can be defined mathematically
- When we see two stars that are close..
  - They may not be close in space
- When we see two stars that appear far...
  - They may not be far in 3-D space
Star Cluster NGC 290 - ESA & NASA

- A picture of the sky is two dimensional
- The stars are not in two dimensions
- When we take a photo of stars we are projecting them into 2-D
  - projecting can be defined mathematically
- When we see two stars that are close...
  - They may not be close in space
- When we see two stars that appear far...
  - They may not be far in 3-D space
Latent Semantic Indexing - Introduction

Star Cluster NGC 290 - ESA & NASA
• When we see two stars that are close in a photo
  • They really are close for some applications
  • For example pointing a big telescope at them
  • Large shared telescopes order their views according to how “close” they are.
Overhead projector example
Overhead projector example

- Depending on where we put the light (and the wall) we can make things in three dimensions appear close or far away in two dimensions.

- Even though the “real” position of the 3-d objects never moved.
Mathematically speaking

• This is taking a 3-D point and projecting it into 2-D

\[
\begin{pmatrix}
10 \\
10 \\
10
\end{pmatrix}
\Rightarrow
\begin{pmatrix}
10 \\
10
\end{pmatrix}
\]

• The arrow in this picture acts like the overhead projector
Mathematically speaking

- We can project from any number of dimensions into any other number of dimensions.
- **Increasing** dimensions adds redundant information
  - But sometimes useful
  - Support Vector Machines (kernel methods) do this effectively
- Latent Semantic Indexing always reduces the number of dimensions
Latent Semantic Indexing - Introduction

Mathematically speaking

- Latent Semantic Indexing always *reduces* the number of dimensions

\[ \begin{pmatrix} 10 \\ 10 \\ 10 \end{pmatrix} \rightarrow \begin{pmatrix} 10 \end{pmatrix} \]
• Latent Semantic Indexing always reduces the number of dimensions.
Mathematically speaking

• Latent Semantic Indexing always reduces the number of dimensions

\[
\begin{pmatrix}
10 \\
10
\end{pmatrix}
\quad \rightarrow \quad
\begin{pmatrix}
10
\end{pmatrix}
\]
Mathematically speaking

- Latent Semantic Indexing always reduces the number of dimensions.
Mathematically speaking

- Latent Semantic Indexing can project on an arbitrary axis, not just a principal axis
Mathematically speaking

- Our documents were just points in an N-dimensional term space
- We can project them also

Antony

Brutus

MacBeth

Othello

Hamlet

Julius Caesar

Antony and Cleopatra

Tempest

Thursday, February 11, 2010
Latent Semantic Indexing - Introduction

Mathematically speaking

- Latent Semantic Indexing makes the claim that these new axes represent **semantics** - deeper meaning than just a term
Mathematically speaking

• A term vector that is projected on new vectors may uncover deeper meanings

• For example

  • Transforming the 3 axes of a term matrix from “ball” “bat” and “cave” to
  
    • An axis that merges “ball” and “bat”
    
    • An axis that merges “bat” and “cave”
  
    • Should be able to separate differences in meaning of the term “bat”
  
• Bonus: less dimensions is faster
Linear Algebra Refresher

- Let C be an M by N matrix with real-valued entries
  - for example our term document matrix
- A matrix with the same number of rows and columns is called a **square matrix**
- An M by M matrix with elements only on the diagonal is called a **diagonal matrix**
- The **identity matrix** is a diagonal matrix with ones on the main diagonal
Let $C$ be an $M$ by $N$ matrix with real-valued entries for example our term document matrix.

A matrix with the same number of rows and columns is called a square matrix.

An $M$ by $M$ matrix with elements only on the diagonal is called a diagonal matrix.

The identity matrix is a diagonal matrix with ones on the main diagonal.

\[
\begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
0 & 2 & 3 & 2 & 1 \\
1 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\]
Linear Algebra Refresher

- Let $C$ be an $M$ by $N$ matrix with real-valued entries, for example our term document matrix.
- A matrix with the same number of rows and columns is called a **square matrix**.
- An $M$ by $M$ matrix with elements only on the diagonal is called a **diagonal matrix**.
- The **identity matrix** is a diagonal matrix with ones on the main diagonal.

\[
C = \begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
0 & 2 & 3 & 2 & 1 \\
1 & 0 & 0 & 1 & 1
\end{bmatrix}
\]
• Let \( C \) be an \( M \) by \( N \) matrix with real-valued entries, for example our term document matrix.

• A matrix with the same number of rows and columns is called a **square matrix**.

• An \( M \) by \( M \) matrix with elements only on the diagonal is called a **diagonal matrix**.

• The **identity matrix** is a diagonal matrix with ones on the main diagonal.
• Let \( C \) be an \( M \) by \( N \) matrix with real-valued entries
  • for example our term document matrix
• A matrix with the same number of rows and columns is called a **square matrix**
• An \( M \) by \( M \) matrix with elements only on the diagonal is called a **diagonal matrix**

• The **identity matrix** is a diagonal matrix with ones on the main diagonal

\[
C = \begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
0 & 2 & 3 & 2 & 1 \\
1 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\]

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

\[
\begin{bmatrix}
5 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 \\
0 & 0 & 5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 7 & 0 \\
0 & 0 & 0 & 0 & 2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
Matrix Decomposition

- Singular Value Decomposition
  - Splits a matrix into three matrices
  - Such that
    - If
    - then
    - and
    - and
    - also Sigma is almost a diagonal matrix
Matrix Decomposition

- Singular Value Decomposition
  - Splits a matrix into three matrices
  - Such that
    - If
    - then
    - and
    - and
    - also Sigma is almost a diagonal matrix
Matrix Decomposition

- **Singular Value Decomposition**
  - Splits a matrix into three matrices
  - Such that
    - If
    - then
    - and
    - and
    - also Sigma is almost a diagonal matrix

\[
C = U \Sigma V^T
\]
Matrix Decomposition

- Singular Value Decomposition
  - Splits a matrix into three matrices
  - Such that
    - If
    - then
    - and
    - and
  - also Sigma is almost a diagonal matrix

\[ C = U \Sigma V^T \]
\[ C \text{ is } (M \text{ by } N) \]
Matrix Decomposition

- Singular Value Decomposition
  - Splits a matrix into three matrices
  - Such that
    - If
    - then
    - and
    - and
    - also Sigma is almost a diagonal matrix

\[ C = U \Sigma V^T \]

\[ C \text{ is } (M \text{ by } N) \]

\[ U \text{ is } (M \text{ by } M) \]
Matrix Decomposition

• Singular Value Decomposition
  • Splits a matrix into three matrices
  • Such that
    • If
    • then
    • and
  • and

  • also Sigma is almost a diagonal matrix
Matrix Decomposition

• Singular Value Decomposition

• Splits a matrix into three matrices

• Such that

• If

• then

• and

• and

• also Sigma is almost a diagonal matrix

\[
C = U \Sigma V^T \\
C \text{ is } (M \text{ by } N) \\
U \text{ is } (M \text{ by } M) \\
\Sigma \text{ is } (M \text{ by } N) \\
V^T \text{ is } (N \text{ by } N)
\]
Matrix Decomposition

\[ C = U \Sigma V^T \]
Matrix Decomposition

- Singular Value Decomposition
  - Is a technique that splits a matrix into three components with these properties.
  - They also have some other properties which are relevant to latent semantic indexing
Matrix Decomposition

- Singular Value Decomposition

- Is a technique that splits a matrix into three components with these properties.
Matrix Decomposition

- **Singular Value Decomposition**

  - SVD enables lossy compression of your term-document matrix
  - reduces the **dimensionality** or the **rank**
  - you can arbitrarily reduce the dimensionality by putting zeros in the bottom right of sigma
  - this is a mathematically optimal way of reducing dimensions
Matrix Decomposition

• Singular Value Decomposition
  
  • If the old dimensions were based on terms
  
  • after reducing the rank of the matrix the dimensionality
  
      is based on concepts or semantics
  
  • a concept is a linear combination of terms

\[
SVD_{\text{dimension}_1} = a \ast td_{\text{dim}_1} + b \ast td_{\text{dim}_2} + c \ast td_{\text{dim}_3} + d \ast td_{\text{dim}_4}
\]
\[
SVD_{\text{dimension}_2} = a' \ast td_{\text{dim}_1} + b' \ast td_{\text{dim}_2} + c' \ast td_{\text{dim}_3} + d' \ast td_{\text{dim}_4}
\]
\[
SVD_{\text{dimension}_3} = a'' \ast td_{\text{dim}_1} + b'' \ast td_{\text{dim}_2} + c'' \ast td_{\text{dim}_3} + d'' \ast td_{\text{dim}_4}
\]
Matrix Decomposition

- Singular Value Decomposition

\[ SVD_{\text{dimension}_1} = a \cdot td_{\text{dim}_1} + b \cdot td_{\text{dim}_2} + c \cdot td_{\text{dim}_3} + d \cdot td_{\text{dim}_4} \]

\[ SVD_{\text{dimension}_2} = a' \cdot td_{\text{dim}_1} + b' \cdot td_{\text{dim}_2} + c' \cdot td_{\text{dim}_3} + d' \cdot td_{\text{dim}_4} \]

\[ SVD_{\text{dimension}_3} = a'' \cdot td_{\text{dim}_1} + b'' \cdot td_{\text{dim}_2} + c'' \cdot td_{\text{dim}_3} + d'' \cdot td_{\text{dim}_4} \]

- 4 dimensions to 3 dimensions

\[
\begin{bmatrix}
a & b & c & d \\
a' & b' & c' & d' \\
a'' & b'' & c'' & d''
\end{bmatrix}
\]
Matrix Decomposition

- Singular Value Decomposition

\[
SVD_{\text{dimension}1} = a \cdot td_{\text{dim}1} + b \cdot td_{\text{dim}2} + c \cdot td_{\text{dim}3} + d \cdot td_{\text{dim}4}
\]

\[
SVD_{\text{dimension}2} = a' \cdot td_{\text{dim}1} + b' \cdot td_{\text{dim}2} + c' \cdot td_{\text{dim}3} + d' \cdot td_{\text{dim}4}
\]

\[
SVD_{\text{dimension}3} = a'' \cdot td_{\text{dim}1} + b'' \cdot td_{\text{dim}2} + c'' \cdot td_{\text{dim}3} + d'' \cdot td_{\text{dim}4}
\]

- 4 dimensions to 3 dimensions

\[
\begin{bmatrix}
  a & b & c & d \\
  a' & b' & c' & d' \\
  a'' & b'' & c'' & d''
\end{bmatrix} \quad \times \quad \begin{bmatrix}
  td_{\text{dim}1} \\
  td_{\text{dim}2} \\
  td_{\text{dim}3} \\
  td_{\text{dim}4}
\end{bmatrix}
\]
Matrix Decomposition

- Singular Value Decomposition

\[
\begin{align*}
SVD_{\text{dimension}_1} &= a \cdot td_{\text{dim}_1} + b \cdot td_{\text{dim}_2} + c \cdot td_{\text{dim}_3} + d \cdot td_{\text{dim}_4} \\
SVD_{\text{dimension}_2} &= a' \cdot td_{\text{dim}_1} + b' \cdot td_{\text{dim}_2} + c' \cdot td_{\text{dim}_3} + d' \cdot td_{\text{dim}_4} \\
SVD_{\text{dimension}_3} &= a'' \cdot td_{\text{dim}_1} + b'' \cdot td_{\text{dim}_2} + c'' \cdot td_{\text{dim}_3} + d'' \cdot td_{\text{dim}_4}
\end{align*}
\]

\[
\begin{array}{cccc}
SVD_{\text{dim}_1} & | & a & b & c & d \\
SVD_{\text{dim}_2} & | & a' & b' & c' & d' \\
SVD_{\text{dim}_3} & | & a'' & b'' & c'' & d''
\end{array}
\]

\[
\begin{array}{c}
\times
\end{array}
\]

\[
\begin{array}{c}
| \hspace{1cm} td_{\text{dim}_1} \\
| \hspace{1cm} td_{\text{dim}_2} \\
| \hspace{1cm} td_{\text{dim}_3} \\
| \hspace{1cm} td_{\text{dim}_4}
\end{array}
\]

Thursday, February 11, 2010
Matrix Decomposition

• Singular Value Decomposition

\[
SVD_{\text{dimension}1} = a \ast td_{\text{dim}1} + b \ast td_{\text{dim}2} + c \ast td_{\text{dim}3} + d \ast td_{\text{dim}4}
\]

\[
SVD_{\text{dimension}2} = a' \ast td_{\text{dim}1} + b' \ast td_{\text{dim}2} + c' \ast td_{\text{dim}3} + d' \ast td_{\text{dim}4}
\]

\[
SVD_{\text{dimension}3} = a'' \ast td_{\text{dim}1} + b'' \ast td_{\text{dim}2} + c'' \ast td_{\text{dim}3} + d'' \ast td_{\text{dim}4}
\]

\[
\begin{bmatrix}
SVD_{\text{dim}1} \\
SVD_{\text{dim}2} \\
SVD_{\text{dim}3}
\end{bmatrix}
= 
\begin{bmatrix}
a & b & c & d \\
a' & b' & c' & d' \\
a'' & b'' & c'' & d''
\end{bmatrix}
\ast
\begin{bmatrix}
td_{\text{dim}1} \\
td_{\text{dim}2} \\
td_{\text{dim}3} \\
td_{\text{dim}4}
\end{bmatrix}
\]

\[
SVD_{\text{ConceptSpace}} = M \ast query_{TermSpace}
\]
Matrix Decomposition

- Singular Value Decomposition

\[
\begin{align*}
SV D_{dim_1} & \quad = \quad \begin{bmatrix} a & b & c & d \\ a' & b' & c' & d' \\ a'' & b'' & c'' & d'' \end{bmatrix} \\
SV D_{dim_2} & \quad = \quad \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} \\
SV D_{dim_3} & \quad = \quad \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} \\
SV D_{ConceptSpace} & \quad = \quad M \times queryTermSpace
\end{align*}
\]

\[
\begin{bmatrix} c \\ \Sigma \\ v^T \end{bmatrix} = \begin{bmatrix} u \\ \vdots \\ \vdots \end{bmatrix}
\]
Matrix Decomposition

- Singular Value Decomposition

\[ \begin{align*}
SV D_{dim_1} & = \begin{bmatrix} a & b & c & d \\ a' & b' & c' & d' \\ a'' & b'' & c'' & d'' \end{bmatrix} \quad \text{and} \\
SV D_{dim_2} & = \begin{bmatrix} t_{d_{dim_1}} \\ t_{d_{dim_2}} \\ t_{d_{dim_3}} \\ t_{d_{dim_4}} \end{bmatrix} \\
SV D_{ConceptSpace} & = M \times query_{TermSpace} \\
M & = \Sigma^{-1} U^T_k \\
c & = u \Sigma V^T
\end{align*} \]
Matrix Decomposition

- Singular Value Decomposition

\[
\begin{bmatrix}
SVD_{dim_1} \\
SVD_{dim_2} \\
SVD_{dim_3}
\end{bmatrix} =
\begin{bmatrix}
a & b & c & d \\
a' & b' & c' & d' \\
a'' & b'' & c'' & d''
\end{bmatrix} \times
\begin{bmatrix}
td_{dim_1} \\
td_{dim_2} \\
td_{dim_3} \\
td_{dim_4}
\end{bmatrix}
\]

\[SVD_{ConceptSpace} = M \times query_{TermSpace}\]

\[M = \Sigma_k^{-1} U_k^T\]

\[query_{ConceptSpace} = \Sigma_k^{-1} U_k^T query_{TermSpace}\]
Matrix Decomposition

- Singular Value Decomposition

- SVD is an algorithm that gives us $\Sigma U V^T$

- With these quantities we can reduce dimensionality

- With reduced dimensionality
  - synonyms are mapped onto the same location
  - “bat” “chiroptera”
  - polysemyes are mapped onto different locations
  - “bat” (baseball) vs. “bat” (small furry mammal)
Latent Semantic Indexing - Linear Algebra Refresher

- Computing SVD takes a significant amount of CPU
- It is possible to add documents to a corpus without recalculating SVD
  - The result becomes an approximation
  - To get mathematical guarantees the whole SVD needs to be computed from scratch
- LSI doesn’t support negation queries
- LSI doesn’t support boolean queries
Matrix Decomposition

- "I am not crazy"
- Netflix
Latent Semantic Indexing - Linear Algebra Refresher

Matrix Decomposition

• “I am not crazy”

• Netflix
Matrix Decomposition

- “I am not crazy”
- Netflix
- Machine translations
  - Just like “bat” and “chiroptera” map to the same
  - “bat” and “murciélago” can map to the same thing
Matrix Decomposition

- "I am not crazy"
- Netflix
- Machine translations
  - Just like "bat" and "chiroptera" map the same
  - "bat" and "murciélago" can map to the same thing

The math is hard but it's beautiful and powerful.
Matrix Decomposition
• “I am not crazy”
• Netflix
• Machine translations
  • Just like “bat” and “chiroptera” map the same
  • “bat” and “murciélago” can map to the same thing

The math is hard but it’s beautiful and powerful
La matemáticas es dura pero es hermosa y de gran alcance
Matrix Decomposition

• “I am not crazy”

• Netflix

• Machine translations
  • Just like “bat” and “chiroptera” map the same
  • “bat” and “murciélago” can map to the same thing

The math is hard but it's beautiful and powerful

La matemáticas es dura pero es hermosa y de gran alcance

Jene mathematisch ist hart, aber ist und an langer Reichweite schön
Matrix Decomposition

• “I am not crazy”
• Netflix
• Machine translations
  • Just like “bat” and “chiroptera” map the same
  • “bat” and “murciélago” can map to the same thing

The math is hard but it’s beautiful and powerful

La matemáticas es dura pero es hermosa y de gran alcance

Jene mathematisch ist hart, aber ist und an langer Reichweite schön

That one mathematically is hard, but is beautiful and at long range