Outline
- Vector Semantics
  - Sparse representation
    - Pointwise Mutual Information (PMI)
  - Dense representation
    - Singular Value Decomposition (SVD)
    - Brown Clusters
    - Neural Language Model

Sparse versus dense vectors
- PPMI vectors are
  - long (length $|V| = 20,000$ to $50,000$)
  - sparse (most elements are zero)

Sparse versus dense vectors
- PPMI vectors are
  - long (length $|V| = 20,000$ to $50,000$)
  - sparse (most elements are zero)
- Alternative: learn vectors which are
  - short (length 200-1000)
  - dense (most elements are non-zero)

Sparse versus dense vectors
- Why dense vectors?
  - Short vectors may be easier to use as features in machine learning (less weights to tune)
  - Dense vectors may generalize better than storing explicit counts
  - They may do better at capturing synonymy:
    - car and automobile are synonyms; but are represented as distinct dimensions; this fails to capture similarity between a word with car as a neighbor and a word with automobile as a neighbor

Three methods for getting short dense vectors
- Singular Value Decomposition (SVD)
- Brown clustering
- “Neural Language Model” — inspired by predictive models
Singular Value Decomposition (SVD)

Rank of a Matrix

• What is the rank of a matrix $A$?

• Number of linearly independent columns of $A$

\[
A = \begin{bmatrix} 1 & 2 & 3 \\ -2 & -3 & 1 \end{bmatrix}
\]

• Rank is 2
• We can rewrite $A$ as two "basis" vectors: $[1 \ 2 \ 1]$ $[-2 \ -3 \ 1]$

Rank as "Dimensionality"

• Rewrite the coordinates in a more efficient way!
  
  • Old basis vectors: $[1 \ 0 \ 0]$, $[0 \ 1 \ 0]$, $[0 \ 0 \ 1]$
  
  • New basis vectors: $[1 \ 2 \ 1]$, $[-2 \ -3 \ 1]$
Intuition of Dimensionality Reduction

- Approximate an N-dimensional dataset using fewer dimensions
- By first rotating the axes into a new space
- In which the highest order dimension captures the most variance in the original dataset
- And the next dimension captures the next most variance, etc.

Sample Dimensionality Reduction

Approximate an N-dimensional dataset using fewer dimensions

Sample Dimensionality Reduction

Any rectangular $w \times c$ matrix $X$ equals the product of 3 matrices:
- $W$: rows corresponding to original but $m$ columns represents a dimension in a new latent space, such that
  - $m$ column vectors are orthogonal to each other
  - Columns are ordered by the amount of variance in the dataset each new dimension accounts for

Singular Value Decomposition

Singular Value Decomposition

Any rectangular $w \times c$ matrix $X$ equals the product of 3 matrices:
- $W$: rows corresponding to original but $m$ columns represents a dimension in a new latent space, such that
  - $m$ column vectors are orthogonal to each other
  - Columns are ordered by the amount of variance in the dataset each new dimension accounts for
- $S$: diagonal $m \times m$ matrix of singular values expressing the importance of each dimension.

Singular Value Decomposition

Singular Value Decomposition

(assuming the matrix has rank $m$)

Landauer and Dumais 1997
**Singular Value Decomposition**

Any rectangular \( w \times c \) matrix \( X \) equals the product of 3 matrices:

\[
X = W \cdot S \cdot C
\]

\( W \): rows to original but \( m \) columns represents a dimension in a new latent space, such that

- \( m \) column vectors are orthogonal to each other
- Columns are ordered by the amount of variance in the dataset each new dimension accounts for

\( S \): diagonal \( m \times m \) matrix of **singular values**

Expressing the importance of each dimension.

\( C \): columns to original but \( m \) rows to singular values

---

**SVD applied to term-document matrix:**

**Latent Semantic Analysis**

Deerwester et al (1988)

- If instead of keeping all \( m \) dimensions, we just keep the top \( k \) singular values.
- Each row of \( W \) (keeping \( k \) columns of the original \( W \)):
  - A \( k \)-dimensional vector
  - Representing word \( w \)

**SVD on Term-Document Matrix: Example**

- The matrix \( X \)

\[
\begin{array}{cccccc}
\text{ship} & 1 & 0 & 1 & 0 & 0 \\
\text{boat} & 0 & 1 & 0 & 0 & 0 \\
\text{ocean} & 1 & 1 & 0 & 0 & 0 \\
\text{wood} & 1 & 0 & 0 & 1 & 1 \\
\text{tree} & 0 & 0 & 0 & 1 & 0 \\
\end{array}
\]

---

**Matrix W**

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
\text{ship} & -0.44 & -0.30 & 0.57 & 0.00 & 0.25 \\
\text{boat} & -0.13 & -0.33 & -0.59 & 0.00 & 0.73 \\
\text{ocean} & -0.48 & -0.51 & -0.57 & 0.00 & 0.51 \\
\text{wood} & -0.70 & 0.35 & 0.13 & -0.58 & 0.16 \\
\text{tree} & 0.26 & 0.65 & -0.41 & 0.58 & -0.10 \\
\end{array}
\]

**Matrix S**

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 & 5 \\
\text{ship} & 0.16 & 0.00 & 0.00 & 0.00 & 0.00 \\
\text{boat} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\text{ocean} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\text{wood} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\text{tree} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\end{array}
\]

**Matrix C**

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
\text{ship} & 0.75 & -0.39 & -0.25 & -0.45 & -0.35 \\
\text{boat} & -0.29 & -0.53 & -0.10 & -0.20 & 0.41 \\
\text{ocean} & 0.58 & 0.89 & 0.01 & -0.05 & -0.58 \\
\text{wood} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\text{tree} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\end{array}
\]

---

**Existing tools from Python, MATLAB, R, etc, for SVD**
Reduce dimension: The Matrix $W$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>-0.44</td>
<td>-0.30</td>
<td>0.57</td>
<td>0.58</td>
<td>0.25</td>
</tr>
<tr>
<td>boat</td>
<td>-0.13</td>
<td>-0.33</td>
<td>-0.59</td>
<td>0.00</td>
<td>0.73</td>
</tr>
<tr>
<td>ocean</td>
<td>-0.48</td>
<td>-0.51</td>
<td>-0.37</td>
<td>0.00</td>
<td>-0.61</td>
</tr>
<tr>
<td>wood</td>
<td>-0.70</td>
<td>0.35</td>
<td>0.15</td>
<td>-0.58</td>
<td>0.16</td>
</tr>
<tr>
<td>tree</td>
<td>-0.26</td>
<td>0.65</td>
<td>-0.41</td>
<td>0.58</td>
<td>-0.09</td>
</tr>
</tbody>
</table>

Similarity between ship and boat vs ship and wood? 

Reduce dimension: The Matrix $S$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>2.16</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>boat</td>
<td>0.00</td>
<td>1.59</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>ocean</td>
<td>0.00</td>
<td>1.28</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>wood</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>tree</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.39</td>
</tr>
</tbody>
</table>

Reduce dimension: The Matrix $C$

<table>
<thead>
<tr>
<th></th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
<th>$d_5$</th>
<th>$d_6$</th>
<th>$d_7$</th>
<th>$d_8$</th>
<th>$d_9$</th>
<th>$d_{10}$</th>
<th>$d_{11}$</th>
<th>$d_{12}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>-0.75</td>
<td>-0.28</td>
<td>-0.20</td>
<td>-0.45</td>
<td>-0.33</td>
<td>-0.12</td>
<td>-0.29</td>
<td>-0.53</td>
<td>-0.19</td>
<td>0.63</td>
<td>0.22</td>
<td>0.41</td>
</tr>
<tr>
<td>boat</td>
<td>-0.53</td>
<td>-0.19</td>
<td>0.63</td>
<td>0.22</td>
<td>0.41</td>
<td>0.00</td>
<td>0.58</td>
<td>0.00</td>
<td>-0.58</td>
<td>0.58</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>ocean</td>
<td>-0.53</td>
<td>-0.29</td>
<td>0.63</td>
<td>0.19</td>
<td>0.41</td>
<td>0.22</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>wood</td>
<td>-0.75</td>
<td>-0.28</td>
<td>-0.20</td>
<td>-0.45</td>
<td>-0.33</td>
<td>-0.12</td>
<td>-0.29</td>
<td>-0.53</td>
<td>-0.19</td>
<td>0.63</td>
<td>0.22</td>
<td>0.41</td>
</tr>
</tbody>
</table>

Reduce dimension: The Matrix $W$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>boat</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ocean</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>wood</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>tree</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Similarity between ship and boat vs ship and wood? 

Reduce dimension: The Matrix $W$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>boat</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ocean</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>wood</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>tree</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
More details

- 300 dimensions are commonly used
- The cells are commonly weighted by a product of two weights (TF-IDF)
  - Local weight: term frequency (or log version)
  - Global weight: idf

Singular Value Decomposition (SVD) is a method for finding the most important singular values, leads to a reduced dataset. $S$ is a diagonal matrix whose entries are the singular values in the original dataset each accounts for. SVD factors a matrix $X$ into a product of three matrices, $W$, $V$, and $C$. Taking only the top $k$ dimensions after SVD applied to co-occurrence matrix $X$:

Let’s return to PPMI word-word matrices

- Can we apply to SVD to them?

Truncated SVD on term-term matrix

$X | V \times | V | = W \begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_\nu \end{bmatrix} C_{k \times |V|}$

(assuming the matrix has rank $|V|$, may not be true)

Truncated SVD produces embeddings

- Each row of $W$ matrix is a $k$-dimensional representation of each word $w$
- $K$ might range from 50 to 1000
- Generally we keep the top $k$ dimensions, but some experiments suggest that getting rid of the top 1 dimension or even the top 50 dimensions is helpful (Lapesa and Evert 2014).
Embeddings versus sparse vectors

• Dense SVD embeddings sometimes work better than sparse PPMI matrices at tasks like word similarity
  • Denoising: low-order dimensions may represent unimportant information
  • Truncation may help the models generalize better to unseen data.
  • Having a smaller number of dimensions may make it easier for classifiers to properly weight the dimensions for the task.
  • Dense models may do better at capturing higher order co-occurrence.