Latent Semantic Indexing

Hoyt Koepke

March 26, 2007

SVD Basics

- Suppose $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x_n}]$ encodes information about our data, with each item represented as a column vector $\mathbf{x}_j$.

SVD Basics

- Suppose $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x_n}]$ encodes information about our data, with each item represented as a column vector $\mathbf{x}_j$.
- Examples
  - Image rasterized as vector (character recognition, etc.).
  - Document represented as a column vector.

SVD Basics

- Suppose $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x_n}]$ encodes information about our data, with each item represented as a column vector $\mathbf{x}_j$.
- Examples
  - Image rasterized as vector (character recognition, etc.).
  - Document represented as a column vector.
- If $\mathbf{U\Sigma V}^T = \mathbf{X}$ is the SVD of $\mathbf{X}$, then there are $r = \text{rank}(\mathbf{X})$ nonzero singular values $\sigma_k \in \mathbf{\Sigma}$. Note that $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$

SVD Basics cont.

- If we set all the singular values past K to be zero (effectively truncating $\mathbf{\Sigma}$), we get a rank K approximation to $\mathbf{X}$.

SVD Basics cont.

- If we set all the singular values past K to be zero (effectively truncating $\mathbf{\Sigma}$), we get a rank K approximation to $\mathbf{X}$.
- Specifically, if $\tilde{\mathbf{X}}$ is the trucated version of $\mathbf{X}$, then

$$\begin{aligned}\tilde{\mathbf{X}} &= \arg\min_{A} \|\mathbf{X} - \mathbf{A}\|_F \\ &= \arg\min_{A} \left[\sum_{i,j} (x_{ij} - a_{ij})^2\right]^{\frac{1}{2}}\end{aligned}$$

Lower Dimensional Representation

- We can then project the columns of $\mathbf{X}$ into an K-dimensional subspace.

$$\tilde{\mathbf{x}}_j = \mathbf{\Sigma}_K^{-1} \mathbf{U}^T \mathbf{x}_j$$

Lower Dimensional Representation

- We can then project the columns of $\mathbf{X}$ into an K-dimensional subspace.

$$\tilde{\mathbf{x}}_j = \mathbf{\Sigma}_K^{-1}\mathbf{U}^T\mathbf{x}_j$$

- We can then query it in this low dimensional subspace.

$$\tilde{\mathbf{q}} = \mathbf{\Sigma}_K^{-1}\mathbf{U}^T\mathbf{q}$$

where $\mathbf{q}$ is the query vector and $\tilde{\mathbf{q}}$ is it's projection into $\mathbf{R}^K$.

Lower Dimensional Representation

- We can then project the columns of $\mathbf{X}$ into an K-dimensional subspace.

$$\tilde{\mathbf{x}}_j = \mathbf{\Sigma}_K^{-1} \mathbf{U}^T \mathbf{x}_j$$

- We can then query it in this low dimensional subspace.

$$\tilde{\mathbf{q}} = \mathbf{\Sigma}_K^{-1} \mathbf{U}^T \mathbf{q}$$

where $\mathbf{q}$ is the query vector and $\tilde{\mathbf{q}}$ is it's projection into $\mathbf{R}^K$.

- Demo.

Vector Representation of Documents

- Given a set of documents $\mathcal{D} = \{d_1, d_2, ..., d_n\}$, with m total terms, we can construct a matrix $\mathbf{X}$ such that each row corresponds to a term and each column corresponds to a document.

Vector Representation of Documents

- Given a set of documents $\mathcal{D} = \{d_1, d_2, ..., d_n\}$, with m total terms, we can construct a matrix $\mathbf{X}$ such that each row corresponds to a term and each column corresponds to a document.
- Each element x_{ij} corresponds to how many times term t_i appears in document d_j.

Latent Semantic Indexing

- One problem with this approach is that terms related in English are independent in the matrix.

Latent Semantic Indexing

- One problem with this approach is that terms related in English are independent in the matrix.
- This makes searching for documents more difficult.

Latent Semantic Indexing

- One problem with this approach is that terms related in English are independent in the matrix.
- This makes searching for documents more difficult.
- For example, suppose someone wanted to find articles on "automobile design," but some of the important articles only mentioned "car", "truck", etc. but not "automobile."

LSI Cont.

- One general property of documents is that related words tend to appear together. If a document mentions a word from a particular subject category, it's more likely that it mentions other words from the same category than mentions unrelated words.

LSI Cont.

- One general property of documents is that related words tend to appear together. If a document mentions a word from a particular subject category, it's more likely that it mentions other words from the same category than mentions unrelated words.
- For example, if you observe that an article mentions "probability", then you'd expect to see words like "distribution," "convergence," etc. more than words like "anemone", "fish", "shark", etc.

LSI Cont.

- One general property of documents is that related words tend to appear together. If a document mentions a word from a particular subject category, it's more likely that it mentions other words from the same category than mentions unrelated words.
- For example, if you observe that an article mentions "probability", then you'd expect to see words like "distribution," "convergence," etc. more than words like "anemone", "fish", "shark", etc.
- We can exploit this using SVD.

SVD of a Document Matrix

- When taking the SVD of a document matrix, documents with similar groups of terms tend to get projected close to each other.

SVD of a Document Matrix

- When taking the SVD of a document matrix, documents with similar groups of terms tend to get projected close to each other.
- This grouping is reflected in the query as well.

SVD of a Document Matrix

- When taking the SVD of a document matrix, documents with similar groups of terms tend to get projected close to each other.
- This grouping is reflected in the query as well.
- Example.

Querying

- To query, we need a measure of document similarity.

Querying

- To query, we need a measure of document similarity.
- Simple distance doesn't work here, because

Querying

- To query, we need a measure of document similarity.
- Simple distance doesn't work here, because
- the query would need to match up the number of occurances as well as the simple existance of words.

Querying

- To query, we need a measure of document similarity.
- Simple distance doesn't work here, because
- the query would need to match up the number of occurances as well as the simple existance of words.
- One simple and highly popular metric is the cosine similarity measure:

$$\mathrm{sim}(\tilde{\mathbf{q}}_1, \tilde{\mathbf{q}}_2) = \frac{\tilde{\mathbf{q}}_1^T \tilde{\mathbf{q}}_2}{||\tilde{\mathbf{q}}_1|| \, ||\tilde{\mathbf{q}}_2||}$$

Examples

- Demo