Information Retrieval
Latent Semantic Indexing

Preliminaries

Vector Space Representation of Documents: TF-IDF

Documents. A single text document is a single unit of retrieval in Information Retrieval systems. Examples of documents are:

- a single paragraph of text
- a tweet
- a news article
- a book
- a book chapter
- a web page
- a transcript of a conversation
- an individual utterance from a conversation

Document collection. A document collection $D = \{d_1, \ldots, d_n\}$ is a set of documents.

Stopwords. A stopword is any word in a language that is considered to carry no important meaning, and that can be ignored when creating feature set representations of documents.
Vocabulary (corpus). The collection of non-stop word words (terms) found in the documents from \( \mathcal{D} \) is called the vocabulary or corpus of \( \mathcal{D} \). Given \( \mathcal{D} \), we denote the vocabulary of \( \mathcal{D} \) as

\[
V_{\mathcal{D}} = \{t_1, \ldots, t_M\}.
\]

(where \( \mathcal{D} \) is unique, we denote the vocabulary of \( \mathcal{D} \) as simply \( V \).) Each \( t_i \) is a distinct term (keyword) found in at least one document in \( \mathcal{D} \).

Bag of words representation. Each document \( d_j \in \mathcal{D} \) is represented as a bag of words, i.e., as an unordered collection of terms found in each document (bag means that the number of occurrences of each term may be taken into account).

The standard representation of bag of words is a vector of keyword weights: a vector which assigns each term \( t_i \in V \) a weight based on its occurrence/non-occurrence in \( d_j \).

As such, we view \( d_j \) as the vector

\[
d_j = (w_{1j}, w_{2j}, w_{3j}, \ldots, w_{Mj}).
\]

Here \( w_{ij} \) is the weight of term \( t_i \) in document \( d_j \).

tf-idf vector space model. The most standard way of modeling text documents and document collections represents keyword weights on the scale from 0 to 1. The keyword weights incorporate two notions: term frequency and inverse document frequency. Cosine similarity to compute the similarity between documents.

Term frequency. Given a document \( d_j \in \mathcal{D} \) and a term \( t_i \in V \), the term frequency (TF) \( f_{ij} \) of \( t_i \) in \( d_j \) is the number of times \( t_i \) occurs in \( d_j \). For a document \( d_j \), we can construct its vector of term frequencies

\[
f_{d_j} = (f_{1j}, f_{2j}, \ldots, f_{Mj}).
\]

Normalized term frequency. Term frequencies are commonly manipulated to provide for a better representation of the document. Two manipulation techniques used are thresholding and normalization.

Given a threshold value \( \alpha \), we set term frequency \( f'_{ij} \) to be

\[
f'_{ij} = \begin{cases} 
   f_{ij} & : f_{ij} < \alpha; \\
   \alpha & : f_{ij} \geq \alpha
\end{cases}
\]

(i.e., we discount any further occurrences of the terms in document beyond a certain threshold \( \alpha \) number of occurrences).

Given a vector \( f_{d_j} \) of (possibly thresholded) term frequencies, we compute normalized term frequencies \( tf_{ij} \) as follows:

\[
 tf_{ij} = \frac{f_{ij}}{\max(f_{1j}, f_{2j}, \ldots, f_{Mj})}.
\]
**Document frequency (DF).** Given a term \( t_i \in V \), its document frequency, \( df_i \) is defined as the number of documents in which \( t_i \) occurs:

\[
df_i = |\{ d_j \in D | f_{ij} > 0 \}|.
\]

**Inverse document frequency (IDF).** Given a term \( t_i \in V \), its inverse document frequency (IDF) is computed as

\[
idf_i = \log \frac{n}{df_i}.
\]

**TF-IDF keyword weighting schema.** Given a document \( d_j \) and a term \( t_i \),

\[
w_{ij} = \frac{f_{ij}}{\max(f_{1j}, \ldots, f_{Mj})} \cdot \log_2 \frac{n}{df_i}.
\]

**Relevance computation.** Vector space retrieval traditionally uses the cosine similarity to compute relevance:

\[
sim(d_j, q) = \cos(d_j, q) = \frac{d_j \cdot q}{||d_j|| \cdot ||q||} = \frac{\sum_{i=1}^{M} w_{ij} \cdot w_{iq}}{\sqrt{\sum_{i=1}^{M} w_{ij}^2 \cdot \sum_{i=1}^{M} w_{iq}^2}}.
\]

**Singular-Valued Decomposition (SVD)**

**Orthogonal Vectors.** Two vectors \( x = (x_1, \ldots, x_M) \) and \( y = (y_1, \ldots, y_M) \) are orthogonal iff

\[
x \cdot y = \sum_{i=1}^{M} x_i \cdot y_i = 0.
\]

**Orthogonal Matrices.** A matrix

\[
V = \begin{pmatrix}
a_{11} & a_{12} & \ldots & a_{1i} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2i} & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
a_{ji} & a_{j2} & \ldots & a_{ji} & \ldots & a_{jn} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \ldots & a_{mi} & \ldots & a_{mn}
\end{pmatrix}
\]

is called orthogonal iff

- each column \( a_i = (a_{1i}, a_{2i}, \ldots, a_{mi}) \) has a length of 1:

\[
\sqrt{\sum_{j=1}^{m} a_{ij}^2} = 1
\]

- vectors in every pair of columns \( a_i, a_k \) for \( 1 \leq i, k \leq n, i \neq k \) are orthogonal:

\[
a_i \cdot a_k = 0
\]
**Lemma.** If $V$ is orthogonal, then

$$VV^T = I,$$

where $I$ is the unit matrix.

**Singular Value Decomposition.** Let $X$ be a matrix

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix}$$

A **Singular Value Decomposition** of $X$ is three matrices $U$, $D$, $V$, such that

$$X = UDV^T,$$

and:

- $V^T$ is an **orthogonal matrix** of size $n \times n$:

$$V^T = \begin{pmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \cdots & v_{nn} \end{pmatrix}$$

- $U$ is an **orthogonal matrix** of size $m \times m$:

$$U = \begin{pmatrix} u_{11} & \cdots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mm} \end{pmatrix}$$

- $D$ is an $m \times n$ matrix, such that $d_{ij} = 0$ for $i \neq j$ and $i = j$, when $i > \min(n, m)$. That is, $D$ is a pseudo-diagonal matrix.

**Theorem.** Any matrix $X$ has a **Singular Value Decomposition**.

**Constructing SVD of a matrix.** We can prove the theorem above by constructing the matrices $U$, $V^T$ and $D$ such that $X = UDV^T$.

**Step 1. Matrix $V$.** Let $V$ be the matrix of principal components of $X$:

$$V = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_q \end{pmatrix}$$

Note: $VV^T = I$, i.e., $V$ is an orthogonal matrix.
Step 2. $XV = UD$. We have matrix $V$ and we have matrix $X$. We can show that $XV$ can be decomposed into an orthogonal matrix $U$ and a diagonal matrix $D$.

Let us compute $S = XV$:

$$s_{ij} = x_i v_j$$

$s_{ij}$ is the score of vector $x_i$ on principal component $v_j$.

Lemma. $s_j = (s_{1j}, \ldots, s_{mj})$ is an eigenvector of $XX^T$.

Proof.

$$s_j = Xv_j$$

$$X^Ts_j = X^TXv_j = \lambda_j v_j$$

$$XX^Ts_j = \lambda_j Xv_j = \lambda_j s_j$$

Because $S$ consists of eigenvectors of $XX^T$, $S$ is orthogonal as well. However, the lengths of the vectors $s_1, \ldots, s_n$ are not necessarily 1 (unlike the lengths of vectors $v_1, \ldots, v_q$).

The lengths of the vectors are

$$||s_j|| = \sqrt{s_j^Ts_j} = \sqrt{v_j^TX^TXv_j} = \sqrt{\lambda_j}$$

This gives us our decomposition:

$$U = \begin{pmatrix} s_1 \sqrt{\lambda_1} & \ldots & s_n \sqrt{\lambda_n} \\ \sqrt{\lambda_1} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sqrt{\lambda_n} \end{pmatrix}$$

$$D = \begin{pmatrix} \sqrt{\lambda_1} & 0 & \ldots & 0 \\ 0 & \sqrt{\lambda_2} & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & \sqrt{\lambda_n} \end{pmatrix}$$

and

$$XV = S = UD$$

Step 3. Finalize

$$XV = UD$$

$$XVV^T = UDV^T$$

$$X = UDV^T$$
SVD matrices. The SVD matrices are:

- \( V \): matrix of principal component vectors of \( X \) (i.e., unit eigenvectors of \( X^T X \). \( V^T \) has eigenvectors in columns.
- \( D \): matrix of singular values: square roots of eigenvectors of \( X^T X \).
- \( U \): matrix of unit eigenvectors of \( XX^T \).

Singular Values. Let \( q = \min(n, m) \). The values \( d_{11}, \ldots, d_{qq} \) from the matrix \( D \) of the singular value decomposition \( X = UDV^T \) are called singular values.

Theorem. Let \( \lambda_1, \ldots, \lambda_q \) be the eigenvalues of matrix \( XX^T \) sorted in descending order. Then, the diagonal elements of matrix \( D \),

\[
d_{ii} = \sqrt{\lambda_i}.
\]

The values \( \sqrt{\lambda_1}, \ldots, \sqrt{\lambda_q} \) are called singular values of matrix \( X \).

Latent Semantic Analysis[1]

Keyword-Term Matrix. Given a document collection \( C = \{d_1, \ldots, d_n\} \), over a vocabulary \( V = \{t_1, \ldots, t_m\} \), where

\[
d_j = (d_{j1}, \ldots, d_{jm}),
\]

the document-by-keyword matrix \( C \) is defined as

\[
C = \begin{pmatrix}
d_{11} & \cdots & d_{1m} \\
\vdots & \ddots & \vdots \\
d_{n1} & \cdots & d_{nm}
\end{pmatrix}
\]

The keyword-by-document matrix \( X = C^T \) is then:

\[
X = C^T = \begin{pmatrix}
d_{11} & \cdots & d_{n1} \\
\vdots & \ddots & \vdots \\
d_{1m} & \cdots & d_{nm}
\end{pmatrix}
\]

\( X \) has \( n \) columns (one column per document) and \( m \) rows (one row per keyword).

Let us apply Singular Value Decomposition to \( X \).

\[
X = UDV^T,
\]

where \( U \) is an orthogonal \( m \times m \) matrix, \( D \) is a pseudo-diagonal \( m \times n \) matrix with singular values of \( XX^T \) on the diagonal, and \( V^T \) is an orthogonal \( n \times n \) matrix.
Note. Let $q = \min(m, n)$. We observe that the above decomposition can be rewritten as

$$X = U_q D_q V_q^T,$$

where $U$ is an orthogonal $m \times q$ matrix, $D$ is a diagonal $q \times q$ matrix with singular values of $XX^T$ on the diagonal, and $V^T$ is an orthogonal $q \times n$ matrix.

Note. In some Information Retrieval problems, the size of the dataset is smaller than the total number of words in them, so $n < m$, or even $n << m$, so, in these situations, $q = n$, and the dimensions of the matrices become: $U : m \times n$, $D : n \times n$, and $V^T : n \times n$.

Approximating $X$. Let $k < q$. The matrix

$$X_k = U_k D_k V_k^T,$$

where

- $U_k$ is the $m \times k$ matrix consisting of the first $k$ columns of matrix $U$
- $D_k$ is a diagonal matrix with $k$ largest singular values $\sigma_1, \ldots, \sigma_k$ of $XX^T$ on the diagonal
- $V_k^T$ is the matrix consisting of the first $k$ columns of matrix $V^T$

Theorem. Matrix $X_k$ is the best rank($k$) matrix approximating $X$.

The approximantion is computed using the Frobenius norm of the matrices:

$$||X|| = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}.$$  

It is known that

$$||X|| = \sqrt{\min(m,n) \sum_{i=1}^{\min(m,n)} \sigma_i^2}.$$  

Interpretations. Matrix $U_k$ has $m$ (number of keywords) rows and $k$ columns. It represents the loadings of the keywords onto the $k$ latent factors.

Matrix $V_k^T$ has $k$ rows and $n$ (number of documents) columns. Each column $v_j$ represents the compacted representation of the document $d_j$ in the space of $k$ latent factors.

Uses. We can now take the matrix $V$ (or $V^T$) and use the rows (columns) instead of the original vectors $d_j = (d_{j1}, \ldots, d_{jm})$ to represent documents.

We can use cosine similarity on the vectors $v_j$ and $v_i$ to find the similarity between two documents in the compacted space.
Query Answering. Suppose a new document (or a query) \( g = (g_1, \ldots, g_m) \) is introduced, and we want to find out how similar \( g \) is to documents in \( D \).

We can obtain the compact representation of \( g \) in our latent space:

\[
v_g = D_k^{-1} U_k^T g.
\]

Note. \( D_k \) is a diagonal matrix, so \( D_k^{-1} \) is a diagonal matrix with values \( \frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_k} \) on the diagonal.

Latent Semantic Indexing: Putting it together. Here is the set of steps to evaluate document similarity in latent space:

1. Construct the keyword-by-document matrix \( X \) for the document collection \( D \).
2. Perform SVD \( X = UDV^T \) on \( X \).
3. Determine \( k < \min(m, n) \): number of latent categories.
4. Extract matrix \( V_k^T \): the first \( k \) rows of matrix \( V^T \).
5. Use columns of \( V_k^T \) as representations of documents.
6. Construct compact representations of other documents by "hitting" them with \( D_k^{-1} U_k^T \) from the left ( \( v_g = D_k^{-1} U_k^T g \)).
7. Use cosine similarity in the space of latent categories to compare documents (old and new) to each other.

References