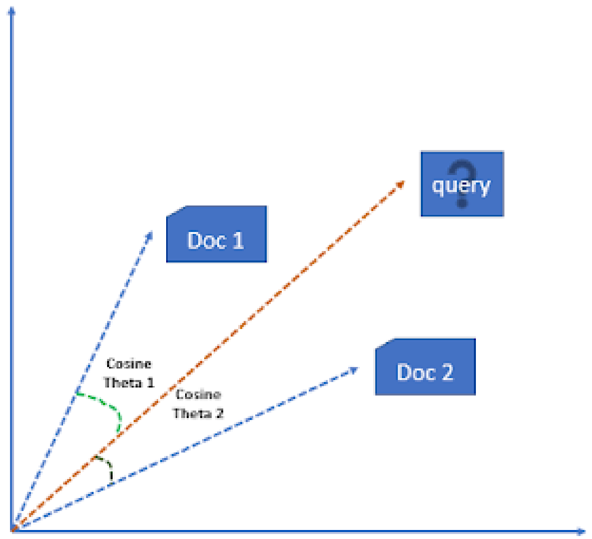
Document Similarity

Natural Language Processing



Agenda

- What is Document Similarity
- Methods to measure Document Similarity
- Cosine Similarity Method

Goal

- Given a set of documents and search term(s)/query we need to retrieve relevant documents that are **similar to the search query**.

Statistical Retrieval

- Retrieval based on similarity between query and documents.
- Output documents are ranked according to similarity to query.
- Similarity based on occurrence frequencies of keywords in query and document.

The Vector-Space Model

- Assume *t* distinct terms remain after preprocessing; call them index terms or the vocabulary.
- These terms form a vector space.
  - Dimensionality = *t* = |vocabulary|
- Each term, *i*, in a document or query, *j*, is given a real-valued weight, w_{ij}.
- Both documents and queries are expressed as *t*-dimensional vectors:
  - $d_j = (w_{1j}, w_{2j}, \ldots, w_{tj})$

Issues for Vector Space Model

- How to determine important words in a document?
  - Word sense?
  - Word n-grams (and phrases, idioms,…)
- How to determine the degree of importance of a term within a document and within the entire collection?
- How to determine the degree of similarity between a document and the query?
- In the case of the web, what is the collection and what are the effects of links, formatting information, etc.?

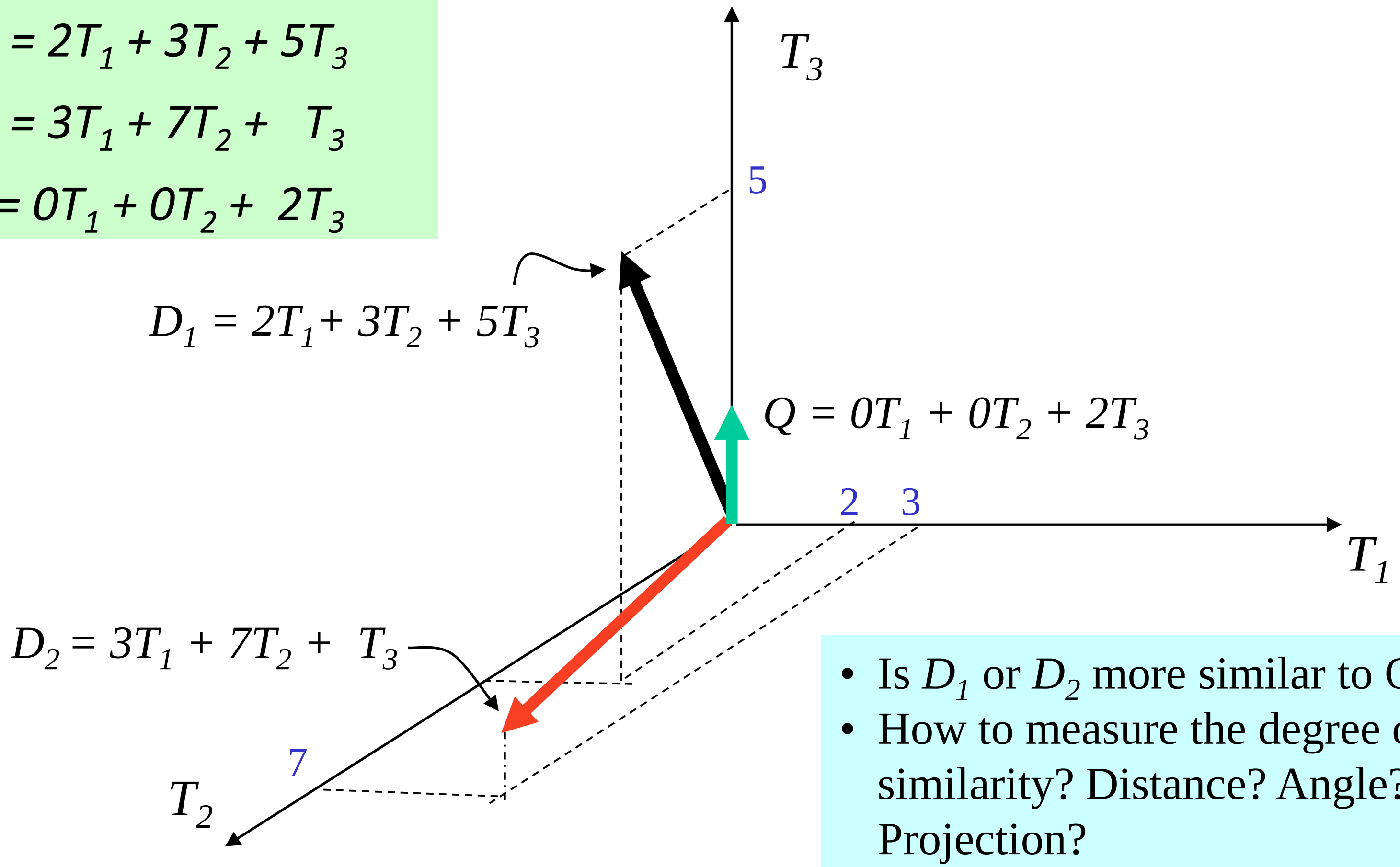
Graphic Representation

Example*:*

$D_1 = 2T_1 + 3T_2 + 5T_3$

$D_2 = 3T_1 + 7T_2 + T_3$

$Q = 0T_1 + 0T_2 + 2T_3$



- Is D_1 or D_2 more similar to Q?
- How to measure the degree of similarity? Distance? Angle? Projection?

Document Collection

A collection of *n* documents can be represented in the vector space model by a term-document matrix.

An entry in the matrix corresponds to the "weight" of a term in the document; zero means the term has no significance in the document or it simply doesn't exist in the document.

$$
\begin{array}{c}
\\ D_1 \\ D_2 \\ \vdots \\ \vdots \\ D_n
\end{array}
\begin{pmatrix}
T_1 & T_2 & \cdots & T_t \\
w_{11} & w_{21} & \cdots & w_{t1} \\
w_{12} & w_{22} & \cdots & w_{t2} \\
\vdots & \vdots & & \vdots \\
\vdots & \vdots & & \vdots \\
w_{1n} & w_{2n} & \cdots & w_{tn}
\end{pmatrix}
$$

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

  f_{ij} = frequency of term i in document j

- May want to normalize *term frequency* (*tf*) by dividing by the frequency of the most common term in the document:

  $tf_{ij} = f_{ij} / max_i\{f_{ij}\}$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

  df_i = document frequency of term i

  = number of documents containing term i

  idf_i = inverse document frequency of term i,

  $= \log_2 (N/ df_i)$

  (N: total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to *tf*.

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N/\, df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

Computing TF-IDF -- An Example

- Given a document containing terms with given frequencies:
- A(3), B(2), C(1)
- Assume collection contains 10,000 documents and
- document frequencies of these terms are:
- A(50), B(1300), C(250)
- Then:
- A: tf = 3/3; idf = log2(10000/50) = 7.6; tf-idf = 7.6
- B: tf = 2/3; idf = log2 (10000/1300) = 2.9; tf-idf = 2.0
- C: tf = 1/3; idf = log2 (10000/250) = 5.3; tf-idf = 1.8

Query Vector

- Query vector is typically treated as a document and also tf-idf weighted.
- Alternative is for the user to supply weights for the given query terms.

Similarity Measure

- A similarity measure is a function that computes the degree of similarity between two vectors.

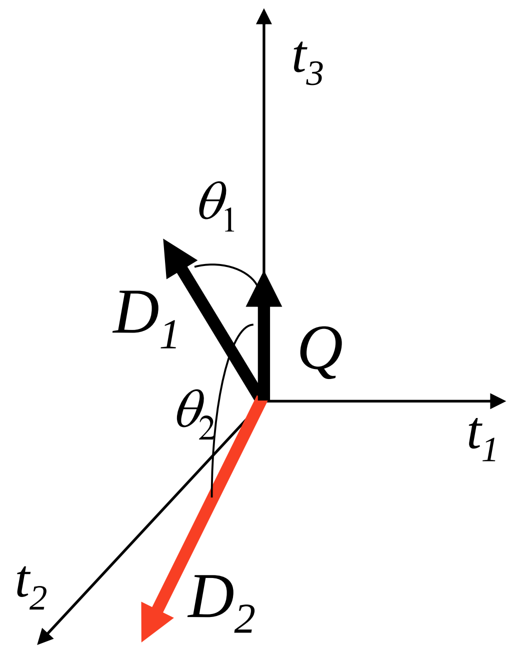
- Using a similarity measure between the query and each document:
- It is possible to rank the retrieved documents in the order of presumed relevance.
- It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.

Cosine Similarity Measure

Cosine similarity measures the cosine of the angle between two vectors.

Inner product normalized by the vector lengths.

$$\text{CosSim}(\mathbf{d}_j, \mathbf{q}) = \frac{\vec{d_j} \cdot \vec{q}}{\left|\vec{d_j}\right| \cdot \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$



$D_1 = 2T_1 + 3T_2 + 5T_3$ CosSim$(D_1, Q) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$

$D_2 = 3T_1 + 7T_2 + 1T_3$ CosSim$(D_2, Q) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$

$Q = 0T_1 + 0T_2 + 2T_3$

D_1 is 6 times better than D_2 using cosine similarity

Naïve Implementation

- Convert all documents in collection D to *tf-idf* weighted vectors, $\boldsymbol{d}_j$, for keyword vocabulary V.
- Convert query to a *tf-idf*-weighted vector $\boldsymbol{q}$.
- For each $\boldsymbol{d}_j$ in D do
- Compute score $s_j = \text{cosSim}(\boldsymbol{d}_j, \boldsymbol{q})$
- Sort documents by decreasing score.
- Present top ranked documents to the user.

- Time complexity: $O(|V|\cdot|D|)$ Bad for large V & D !
- $|V| = 10{,}000$; $|D| = 100{,}000$; $|V|\cdot|D| = 1{,}000{,}000{,}000$

Comments on Vector Space Models

- Simple, mathematically based approach.
- Considers both local (tf) and global (idf) word occurrence frequencies.
- Provides partial matching and ranked results.
- Tends to work quite well in practice despite obvious weaknesses.
- Allows efficient implementation for large document collections.

Problems with Vector Space Model

- Missing semantic information
  - word sense
- Missing syntactic information
  - phrase structure, word order, proximity information
- Assumption of term independence
  - ignores synonomy
- Given a two-term query “A B”
  - may prefer a document containing A frequently but not B, over a document that contains both A and B, but both less frequently.