63

Information Retrieval and Web Search

Amy N. Langville
The College of Charleston

Carl D. Meyer
North Carolina State University

Information retrieval is the process of searching within a document collection for information most relevant to a user's query. However, the type of document collection significantly affects the methods and algorithms used to process queries. In this chapter, we distinguish between two types of document collections: traditional and Web collections. Traditional information retrieval is search within small, controlled, nonlinked collections (e.g., a collection of medical or legal documents), whereas Web information retrieval is search within the world's largest and linked document collection. In spite of the proliferation of the Web, more traditional nonlinked collections still exist, and there is still a place for the older methods of information retrieval.

63.1 The Traditional Vector Space Method

Today most search systems that deal with traditional document collections use some form of the vector space method [SB83] developed by Gerard Salton in the early 1960s. Salton's method transforms textual data into numeric vectors and matrices and employs matrix analysis techniques to discover key features and connections in the document collection.

Definitions:

For a given collection of documents and for a dictionary of m terms, document i is represented by an $m \times 1$ **document vector** $\mathbf{d}_i$ whose jth element is the number of times term j appears in document i.

The **term-by-document matrix** is the $m \times n$ matrix

$$A = [\, \mathbf{d}_1 \, \mathbf{d}_2 \, \cdots \, \mathbf{d}_n \,]$$

whose columns are the document vectors.

Recall is a measure of performance that is defined to be

$$0 \leq \text{Recall} = \frac{\#\text{ relevant docs retrieved}}{\#\text{ relevant docs in collection}} \leq 1.$$

Precision is another measure of performance defined to be

$$0 \leq \text{Precision} = \frac{\#\text{ relevant docs retrieved}}{\#\text{ docs retrieved}} \leq 1.$$

Query processing is the act of retrieving documents from the collection that are most related to a user's query, and the **query vector** $\mathbf{q}_{m\times 1}$ is the binary vector defined by

$$q_i = \begin{cases} 1 & \text{if term } i \text{ is present in the user's query,} \\ 0 & \text{otherwise.} \end{cases}$$

The **relevance** of document i to a query $\mathbf{q}$ is defined to be

$$\delta_i = cos\ \theta_i = \mathbf{q}^T\mathbf{d}_i / \|\mathbf{q}\|_2 \|\mathbf{d}_i\|_2.$$

For a selected tolerance τ, the **retrieved documents** that are returned to the user are the documents for which $\delta_i > \tau$.

Facts:

1. The term-by-document matrix A is sparse and nonnegative, but otherwise unstructured.
2. [BB05] In practice, weighting schemes other than raw frequency counts are used to construct the term-by-document matrix because weighted frequencies can improve performance.
3. [BB05] Query weighting may also be implemented in practice.
4. The tolerance τ is usually tuned to the specific nature of the underlying document collection.
5. Tuning can be accomplished with the technique of relevance feedback, which uses a revised query vector such as $\tilde{\mathbf{q}} = \delta_1\mathbf{d}_1 + \delta_3\mathbf{d}_3 + \delta_7\mathbf{d}_7$, where $\mathbf{d}_1$, $\mathbf{d}_3$, and $\mathbf{d}_7$ are the documents the user judges most relevant to a given query $\mathbf{q}$.
6. When the columns of A and $\mathbf{q}$ are normalized, as they usually are, the vector $\boldsymbol{\delta}^T = \mathbf{q}^T A$ provides the complete picture of how well each document in the collection matches the query.
7. The vector space model is efficient because A is usually very sparse, and $\mathbf{q}^T A$ can be executed in parallel, if necessary.
8. [BB05] Because of linguistic issues such as polysomes and synonyms, the vector space model provides only decent performance on query processing tasks.
9. The underlying basis for the vector space model is the standard basis $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m$, and the orthogonality of this basis can impose an unrealistic independence among terms.
10. The vector space model is a good starting place, but variations have been developed that provide better performance.

Examples:

1. Consider a collection of seven documents and nine terms (taken from [BB05]). Terms not in the system's index are ignored. Suppose further that only the titles of each document are used for indexing. The indexed terms and titles of documents are shown below.

Terms	Documents
T1: Bab(y,ies,y's)	D1: *Infant* & *Toddler* First Aid
T2: Child(ren's)	D2: *Babies* and *Children's* Room (For Your *Home*)
T3: Guide	D3: *Child Safety* at *Home*
T4: Health	D4: Your *Baby's Health* and *Safety*: From *Infant* to *Toddler*
T5: Home	D5: *Baby Proofing* Basics
T6: Infant	D6: Your *Guide* to Easy Rust *Proofing*
T7: Proofing	D7: Beanie *Babies* Collector's *Guide*
T8: Safety	
T9: Toddler	

The indexed terms are italicized in the titles. Also, the stems [BB05] of the terms for baby (and its variants) and child (and its variants) are used to save storage and improve performance. The term-by-document matrix for this document collection is

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

For a query on *baby health*, the query vector is

$$\mathbf{q} = [\,1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\,]^T.$$

To process the user's query, the cosines

$$\delta_i = cos\,\theta_i = \frac{\mathbf{q}^T \mathbf{d}_i}{\|\mathbf{q}\|_2 \|\mathbf{d}_i\|_2}$$

are computed. The documents corresponding to the largest elements of $\boldsymbol{\delta}$ are most relevant to the user's query. For our example,

$$\boldsymbol{\delta} \approx [\,0 \quad 0.40824 \quad 0 \quad 0.63245 \quad 0.5 \quad 0 \quad 0.5\,],$$

so document vector 4 is scored most relevant to the query on *baby health*. To calculate the recall and precision scores, one needs to be working with a small, well-studied document collection. In this example, documents $\mathbf{d}_4$, $\mathbf{d}_1$, and $\mathbf{d}_3$ are the three documents in the collection relevant to baby health. Consequently, with $\tau = .1$, the recall score is $1/3$ and the precision is $1/4$.

63.2 Latent Semantic Indexing

In the 1990s, an improved information retrieval system replaced the vector space model. This system is called Latent Semantic Indexing (LSI) [Dum91] and was the product of Susan Dumais, then at Bell Labs. LSI simply creates a low rank approximation A_k to the term-by-document matrix A from the vector space model.

Facts:

1. [Mey00] If the term-by-document matrix $A_{m\times n}$ has the singular value decomposition $A = U\Sigma\ V^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, then A_k is created by truncating this expansion after k terms, where k is a user tunable parameter.
2. The recall and precision measures are generally used in conjunction with each other to evaluate performance.
3. A is replaced by $A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ in the query process so that if $\mathbf{q}$ and the columns of A_k have been normalized, then the angle vector is computed as $\boldsymbol{\delta}^T = \mathbf{q}^T A_k$.
4. The truncated SVD approximation to A is optimal in the sense that of all rank-k matrices, the truncated SVD A_k is the closest to A, and

$$\|A - A_k\|_F = \min_{rank(B)\leq k} \|A - B\|_F = \sqrt{\sigma_{k+1}^2 + \cdots + \sigma_r^2}.$$

5. This rank-k approximation reduces the so-called linguistic noise present in the term-by-document matrix and, thus, improves information retrieval performance.
6. [Dum91], [BB05], [BR99], [Ber01], [BDJ99] LSI is known to outperform the vector space model in terms of precision and recall.
7. [BR99], [Ber01], [BB05], [BF96], [BDJ99], [BO98], [Blo99], [BR01], [Dum91], [HB00], [JL00], [JB00], [LB97], [WB98], [ZBR01], [ZMS98] LSI and the truncated singular value decomposition dominated text mining research in the 1990s.
8. A serious drawback to LSI is that while it might appear at first glance that A_k should save storage over the original matrix A, this is often not the case, even when $k << r$. This is because A is generally very sparse, but the singular vectors $\mathbf{u}_i$ and $\mathbf{v}_i^T$ are almost always completely dense. In many cases, A_k requires more (sometimes much more) storage than A itself requires.
9. A significant problem with LSI is the fact that while A is a nonnegative matrix, the singular vectors are mixed in sign. This loss of important structure means that the truncated singular value decomposition provides no textual or semantic interpretation. Consider a particular document vector, say, column 1 of A. The truncated singular value decomposition represents document 1 as

$$A_1 = \begin{bmatrix} \vdots \\ \mathbf{u}_1 \\ \vdots \end{bmatrix} \sigma_1 v_{11} + \begin{bmatrix} \vdots \\ \mathbf{u}_2 \\ \vdots \end{bmatrix} \sigma_2 v_{12} + \cdots \begin{bmatrix} \vdots \\ \mathbf{u}_k \\ \vdots \end{bmatrix} \sigma_k v_{1k},$$

   so document 1 is a linear combination of the basis vectors $\mathbf{u}_i$ with the scalar $\sigma_i v_{1i}$ being a weight that represents the contribution of basis vector i in document 1. What we'd really like to do is say that basis vector i is mostly concerned with some subset of the terms, but any such textual or semantic interpretation is difficult (or impossible) when SVD components are involved. Moreover, if there were textual or semantic interpretations, the orthogonality of the singular vectors would ensure that there is no overlap of terms in the topics in the basis vectors, which is highly unrealistic.
10. [Ber01], [ZMS98] It is usually a difficult problem to determine the most appropriate value of k for a given dataset because k must be large enough so that A_k can capture the essence of the document collection, but small enough to address storage and computational issues. Various heuristics have been developed to deal with this issue.