Text Retrieval using Linear Algebra

Joshua Drew

Josh Drew graduated summa cum laude in 2002 from Ball State with a B.S. degree in Computer Science and a minor in Physics. This article is based on his research during 2001 and 2002. Josh is currently a senior programmer with SpinWeb Net Designs, Inc.

Text retrieval is an important area of research. As information and methods of its storage have proliferated, the need to have efficient methods of locating subsets of this information has increased as well. A widely-researched text searching method involves modeling a text collection in a term-by-document matrix, and evaluating the documents' relevance to a query with simple linear algebra. This document is an abstract of research performed throughout 2001 and 2002, and presents one such system, developed to search the bsu-cs webserver.

In this method, a matrix A is constructed such that the frequency of each term i in each document j, is stored at $A_{i,j}$. To build this matrix for bsu-cs, a simple index-creator was programmed. Beginning with a list of known Web pages, the script analyzed each document encountered. Word frequencies were recorded, and links to other bsu-cs Web pages were extracted. Those links were both added to the page list, and tabulated as possible indicators of document importance (more links to page x implies that page x is more important). Note that common words such as "the" and "how" have little intrinsic meaning [1]. Therefore, a list of "stop words" was created. The members of this list were excluded by the matrix-forming code. After several hours of parsing, the index-creator, in conjunction with a script to organize each page's raw data, produced a 26257 by 3557, sparse matrix.

If a user supplies a list of key words, the relevance of those words to each of the documents in the term-by-document matrix $A = [\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n]$ can be determined. A common measure of such is the cosine of the angle between the query vector, $\vec{q}$ and each document (column) vector, $\vec{a}_j$ [3]. If the query is represented by a vector $\vec{q}$ and the term-by-document matrix as A, the cosine of the angle between $\vec{q}$ and a document $\vec{a}_j$ is found by:

$$\cos\theta_j = (\vec{a}_j \cdot \vec{q})/(\|\vec{a}_j\|_2 \|\vec{q}\|_2); \qquad j = 1, 2, \ldots, n$$

where the Euclidean vector norm $\|\vec{x}\|_2$ is defined by $\|\vec{x}\|_2 = (\vec{x} \cdot \vec{x})^{1/2}$, the square root of the dot product of $\vec{x}$ with itself [3]. The results of this computation can be stored for all documents, $j = 1, \ldots, n$. If the cosine of the angle between two vectors is one, the vectors are parallel. However, if the cosine of the angle is zero, the vectors are orthogonal [6]. Therefore, a cosine closer to one implies that a document, $\vec{a}_j$, is relevant to a user's search vector, $\vec{q}$. In general, documents not exceeding some cosine threshold, which is determined experimentally, are returned to the user [4]. The C++ program, "search"[1], uses this idea, and a threshold of 0.132, to provide users with query results.

The rank of a matrix is the dimension of the column space of that matrix [6]. So, for the term-by-document matrix A, a rank-reduction involves removing unnecessary documents, e.g. a Web page mirror. These are eliminated with the consequence of improved search results.

This project uses a truncated singular value decomposition (SVD) [5] to approximate A.

Let A be an m by n matrix. Then a SVD of A always exists and is given by $A = USV^T$ with an m by m orthogonal matrix U, an m by n matrix S, and an n by n orthogonal matrix V, such that the only non-zero entries of S are along its diagonal and are non-increasing and non-negative (they are called the *singular values*). Focusing on the first k singular values, we obtain a truncated SVD if we take only the first k columns of U, the k by k submatrix of S containing our selected singular values, and the first k columns of V. That is, we can approximate $A \approx USV^T$, where U is an m by k matrix with orthonormal columns, S a k by k diagonal matrix, and V an n by k matrix with orthonormal columns.

Several benefits arise from this usage:

- Noise and uncertainty, present in all large databases, are reduced [1].
- Word usage is estimated across documents [2], helping to compensate for polysemy, the situation in which a word has multiple meanings.
- Queries may return relevant documents containing none of the user's search terms. The SVD models a latent semantic structure assumed to exist in the document collection [2].
- Calculations can be performed faster because A is replaced by a SVD approximation.

Advanced information retrieval methods are crucial to making today's large text collections useful. This project establishes a foundation on which students at the undergraduate or graduate level can explore new possibilities such as:

- Developing new weighting schemes based upon term and document importance and the likelihood of relevance to a user's query.
- Using an adjacency matrix of documents to rank pages; taking advantage of the fact that the number of k-step sequences between vertex i and j in a graph with the adjacency matrix M is the (i, j) entry in M^k.

[1] The source code for "search" can be found on the project Web site

- Modularizing server-side processes by separating this project's search-performing program into a client and a daemon which communicate via sockets.

This project produced a working search engine for the bsu-cs sever. All relevant source code, input, output, and documentation produced is available at `http://www.joshdrew.com/thesis/`

My sincere thanks go to Dr. James Baglama, my faculty advisor, and Scott Hinkley for their invaluable contributions to this project.

References

[1] M.W. Berry, M. Browne, Understanding search engines: mathematical modeling and text retrieval, SIAM (1999).

[2] M.W.Berry, S. T. Dumais, G.W. O'Brien, *Using linear algebra for intelligent information retrieval*, SIAM Rev. **37** (1995) 573–595.

[3] M.W. Berry, Z. Drmac, E.R. Jessup, *Matrices, vector spaces, and information retrieval*, SIAM Rev. **41** (2) 335–362.

[4] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, R. Harshman, *Indexing by latent semantic analysis*, Journal of the American Society for Information Science **41** (1990) 391–407.

[5] G. Golub, C.V. Loan, Matrix computations (Second Ed.), Johns-Hopkins University Press (1989).

[6] D.C. Lay, Linear algebra and its applications (Second Ed.), Addison-Wesley (2000).