Linear Algebra for Information Retrieval

Rishabh Gupta, Salman Baqri, Nilanjan Debnath, Vanshi Mishra

25.04.2020

Introduction

The advent of Internet and subsequent widespread access to it has dramatically changed the way in which we create, process and manage documents online. Both, the scale and the methods of storing and querying documents poses a unique challenge today.

The manual methods of indexing are succumbing to problems of both capacity and consistency. There are nearly 1.4 million books in print in the United States alone, with approximately 60,000 new titles appearing there annually. The experience and opinions of the indexer can significantly affect the extraction and documentation of key words. Experiments have shown that there is a 20% disparity on average in the terms chosen as appropriate to describe a given document by two different professional indexers. Automated methods of Information retrieval arrived to conquer these specific challenges. However these methods have their own problems, complexities of language being one of them(polysemy and synonymy).

In Vector Space Model, documents are stored as vectors whose components determine the importance of that term in reflecting its semantics. The document vectors are stored as columns of a matrix called term-document matrix. The user's query is modelled as a vector and compared to the term-document matrix using cosine similarities. The new document or a new term can be added by appending the matrix with a new column or new row respectively. QR factorization and SVD is then used for geometric interpretation of vector space model and for rank reduction of the term-document matrix to account for the uncertainties in the database. The two factorization methods are then compared for their specific benefits.

Objective

The objective of this project is to implement Vector space model, QR-Factorization and SVD on a dataset for Information Retrieval.

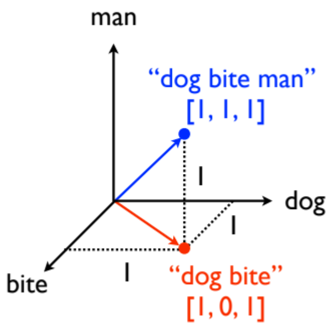
- Vector Space Model- To provide a vector space representation of dataset. To perform query matching.

- QR Factorization- To identify and remove redundant information from term-document matrix.
- SVD- For low rank approximation of the matrix and Query matching.
- Term-Term Comparison.

Vector Space Model

We will be representing each document in the collection as a vector. The value assigned to the components reflects the importance of the term in representing the semantics of the document. The components of the vector are selected after stemming and removing stock words and punctuations.

Let us have 2 documents :
doc_1 =“dog bite man”
which can be represented as the vector [1,1,1]
doc_2 =“dog bite”
which can be represented as the vector [1.0.1]
This can be represented in the Vector Space as:



A database containing a total of d documents described by t terms is represented as a $t \times d$ term-by-document matrix A. The d vectors representing the d documents form the columns of the matrix. Thus, the matrix element a_{ij} is the weighted frequency at which term i occurs in document j. The elements a_{ij} of the term-by-document matrix A are often assigned two-part values $a_{ij} = l_{ij} g_i$. In this case, the factor g_i is a global weight that reflects the overall value of term i as an indexing term for the entire collection.

The term document matrix for the above example is:

	doc_1	doc_2
dog	1	1
man	1	0
bite	1	1

Our aim is to have a measure of similarity among documents and between the query and document vector. For this purpose we are using cosine similarity. Suppose that we want to match a query vector to a document, then that document would be represented by jth column of term-document matrix and the similarity measure will be

$$\cos\theta_j = \frac{a_j^T q}{\|a_j\|_2 \|q\|} = \frac{\sum_{i=1}^{t} a_{ij} q_i}{\sqrt{\sum_{i=1}^{t} a_{ij}^2}\sqrt{\sum_{i=1}^{t} q_i^2}} \tag{1}$$

For example, the cosine similarity between the two documents in above example is $\frac{\sqrt{2}}{\sqrt{3}}$

We observe that there is a lot of redundant data in term-document matrix. For this we need to identify the rank of the matrix and apply rank reduction. This step allows us o set portions of the matrix to zero and to ignore them in the subsequent computations. One way to do this is by using QR factorisation.

QR Factorization

Basis for Column Space

Any matrix A can be decomposed into a product of Q (orthogonal matrix) and R(upper triangular matrix).$A = QR$ simply that columns of A are linear combinations of columns of Q. Therefore, a subset of columns of Q are the basis of $c(A)$. Let r be the rank of A and if we use column pivoting for factorization then first r columns of Q will be the basis for column space of A.

$$A = \begin{pmatrix} 0.5774 & 0 & 0 & 0.4082 & 0 \\ 0.5774 & 0 & 1 & 0.4082 & 0.7071 \\ 0.5774 & 0 & 0 & 0.4082 & 0 \\ 0 & 0 & 0 & 0.4082 & 0 \\ 0 & 1 & 0 & 0.4082 & 0.7071 \\ 0 & 0 & 0 & 0.4082 & 0 \end{pmatrix} \tag{2}$$

$$Q = \left(\begin{array}{cccc|cc} -0.5774 & 0 & -0.4082 & 0 & -0.7071 & 0 \\ -.5774 & 0 & 0.8165 & 0 & 0 & 0 \\ -0.5774 & 0 & -0.4082 & 0 & 0.7071 & 0 \\ 0 & 0 & 0 & -0.7071 & 0 & -0.7071 \\ 0 & -1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.7071 & 0 & 0.7071 \end{array}\right)$$

$$R = \left(\begin{array}{ccccc} -1.0001 & 0 & -0.5774 & -0.7070 & -0.4082 \\ 0 & -1.0000 & 0 & -0.4082 & -0.7071 \\ 0 & 0 & 0.8165 & 0 & 0.5774 \\ 0 & 0 & 0 & -0.5774 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

Let

$$\mathrm{Q}_A = \begin{pmatrix} -0.5774 & 0 & -0.4082 & 0 \\ -.5774 & 0 & 0.8165 & 0 \\ -0.5774 & 0 & -0.4082 & 0 \\ 0 & 0 & 0 & -0.7071 \\ 0 & -1.0000 & 0 & 0 \\ 0 & 0 & 0 & -0.7071 \end{pmatrix}$$

$$\mathrm{R}_A = \begin{pmatrix} -1.0001 & 0 & -0.5774 & -0.7070 & -0.4082 \\ 0 & -1.0000 & 0 & -0.4082 & -0.7071 \\ 0 & 0 & 0.8165 & 0 & 0.5774 \\ 0 & 0 & 0 & -0.5774 & 0 \end{pmatrix}$$

This can be used to identify and remove the redundant information from the database. $A = QR$ can be re-written as

$$A = \begin{bmatrix} Q_A & Q_A^{\perp} \end{bmatrix} \begin{bmatrix} R_A \\ 0 \end{bmatrix}$$

$$A = Q_A R_A + Q_A^{\perp} \cdot 0 = Q_A R_A \tag{3}$$

With column pivoting, the first r columns of Q form a basis for the column space of the rank r matrix A, and the elements of the first r rows of R provide the coefficients for the linear combinations of those basis vectors that constitute the columns of A.The contents of database will be fully described by the columns of Q_A. The query matching will be described by the cosine similarities between a query vector q and and columns of Q_A. If the document is column a then cosine will be given by

$$\cos\theta_j = \frac{a_j^T q}{\|a_j\|_2 \|q\|} = \frac{(Q_A r_j)^T q}{\|Q_A r_j\|_2 \|q\|_2} = \frac{r_j^T (Q_A^T q)}{\|r_j\|_2 \|q\|_2} \tag{4}$$

where r_j is the jth column of R and a_j is the jth column of As

Geometry of a Vector Space Model

We can write the query vector q as the sum of its projection the column space of A and in the orthogonal complement of column space.

$$
\begin{aligned}
q = Iq &= QQ^T \\
&= [\mathrm{Q}_A Q_A^T + Q_A^{\perp}(Q_A^{\perp T}]q \\
&= \mathrm{Q}_A Q_A^T q + Q_A^{\perp}(Q_A^{\perp T} q \\
&= \mathrm{q}_A + q_A^{\perp}
\end{aligned}
$$

The projection in column space is called orthogonal projection. This projection is the closest approximation of Q in column space.

Low Rank Approximation

The process of indexing the database can also lead to uncertainties in the term-document matrix. The indexing could have been performed by various people over a large period of time. The translation of data into matrix is subject to interpretation therefore the term-document could be represented by another matrix $A+E$. The uncertainty matrix E reflects incomplete information about the database.

Let the rank of A is r then assume A+E has a lower rank. The aim is to find a reasonable low rank approximation of A. From $A = QR$, we can write :-

$$
R = \left(\begin{array}{ccc|cc}
-1.0001 & 0 & -0.5774 & -0.7070 & -0.4082 \\
0 & -1.0000 & 0 & -0.4082 & -0.7071 \\
0 & 0 & 0.8165 & 0 & 0.5774 \\
\hline
0 & 0 & 0 & -0.5774 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{array}\right)
$$

$$
= \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}
$$

The QR factorization with column pivoting tend to separate large and small values of R. It pushes the larger values to the top right corner and zeros to the opposite. We change R_{22} to 0 and the R changes to Z(say). The rank of Z has reduced now. Write $A+E = QZ$. The uncertainty matrix is given by $E = (A+E) - A$

$$
\|E\|_F = \|R_{22}\|_F \tag{5}
$$

For example:- Because $\|A\|_F = \|R\|_F$, $\|E\|_F/\|A\|_F = \|R_{22}\|_F/\|R\|_F = 0.2582$.In words, making a 26% relative change in the value of R makes the same sized change in A, and that change reduces the ranks of both matrices by one.

The relative changes made in R made the same change in A and it reduces the rank of both the matrix. Examples shows that by choosing suitable R_{22} uncertainties made in A can roughly be same as made by simply choosing different indexer(Since in the introduction of the project it was given that two different indexer can have 20% acuracy on term-document matrix).

Singular Value Decomposition

The singular value decomposition of any matrix gives $A = U\Sigma V$

The columns of U and V are respectively called left and right singular vectors of A. U and V are orthogonal matrices. S is a diagonal matrix which contains singular values of A in decreasing as we move downward on its diagonal. There are many parallels between $A = QR$ and $A = U\Sigma V$. Number of non zero elements on diagonals of Q and S give the rank of A. In both cases first r columns of Q and U form the basis for column space of A. But in addition first r rows of transpose of V form the basis of row space of A. The rank k approximation of A can also be created by setting all but k largest singular values to zero. The difference between QR and SVD factorization is that K rank approximation matrix obtained by SVD is closer to A then any other A_k.

$$\|A\|_F = \|U\Sigma^T\|_F = \|\Sigma\|_F = \sqrt{\sum_{j=1}^{r_A} \sigma_j^2}$$

$$\text{A}_k = U_k \Sigma_k V_K^T$$

Where U_k is the $t \times k$ matrix whose columns are first k columns of U. V_K is the $d \times k$ matrix whose columns are the first k columns of V and Σ_k is the $k \times k$ diagonal matrix whose diagonal elements are the k largest singular values of A.

By Eckart-Young Formula, the distance between A and A_k is minimized by this approximation. Also,

$$\|A - A_k\| = \sqrt{\sigma_{k+1}^2 + ... + \sigma_{rA}^2}$$

Returning to our example matrix A, we have $A = U\Sigma V^T$ where,

$$U = \begin{pmatrix} 0.2670 & 0.2567 & 0.5308 & -0.22847 & -0.7071 & 0 \\ 0.7479 & -0.3981 & -0.5249 & 0.0816 & 0 & 0 \\ 0.2670 & -0.2567 & 0.5308 & -0.2847 & 0.7071 & 0 \\ 0.1182 & -0.0127 & 0.2774 & 0.6394 & 0 & -0.7071 \\ 0.5198 & 0.8423 & 0.0838 & -0.1158 & 0 & 0 \\ 0.1182 & -0.0127 & 0.2774 & 0.6394 & 0 & 0.7071 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1.6950 & 0 & 0 & 0 & 0 \\ 0 & 1.1158 & 0 & 0 & 0 \\ 0 & 0 & 0.8403 & 0 & 0 \\ 0 & 0 & 0 & 0.4195 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$V = \begin{pmatrix} 0.4366 & -0.4717 & 0.3688 & -0.6715 & 0 \\ 0.3067 & 0.7549 & 0.0998 & -0.2760 & -0.5000 \\ 0.4412 & -0.3568 & -0.6247 & 0.1945 & -0.5000 \\ 0.4909 & -0.0346 & 0.5711 & 0.6571 & 0 \\ 0.5288 & 0.2815 & -0.3712 & -0.0577 & 0.7071 \end{pmatrix}$$

The relative change of 19% is required in SVD to reduce rank by 1 as compared to corresponding change of 26% in QR factorization.

Reduced-Rank Vector Space Model

We can formulate query matching from SVD as well. We compare a query vector q to the columns of the approximation A_k . Let e_j be the jth column of $d * d$ identity matrix, then jth column of A_k can be given by $A_k e_j$. The cosine for the angle between query vector an approximated documents will be given by the formula

$$\cos\theta_j = \frac{(A_k e_j)^T q}{\|A_k e_j\|_2 \|q\|_2} = \frac{(U_k \Sigma_k V_k^T e_j)^T q}{\|U_k \Sigma_k V_k^T e_j\|_2 \|q\|_2} = \frac{e_j^T V_k \Sigma_k (U_k^T q)}{\|\Sigma_k V_k^T e_j\|_2 \|q\|_2} \qquad (6)$$

Term-Term Comparison

As the name suggests like term document comparison we can also compare terms with each other using cosine similarity. Let the angle between two terms vectors i and j be ω_{ij} . The cosine of this angle between two vectors can be given by

$$\cos\omega_{ij} = \frac{(e_i^T G)(G^T e_j)}{\|G^T e_i\|_2 \|G^T e_j\|_2} \qquad (7)$$

Where G is the term by document matrix and e_l denotes the lth canonical vector of dimension t.

Cosines of angles between term vectors in our used example :

$$C = \left(\begin{array}{c|ccc|cc|c} 1.0000 & 0.3464 & 0.3464 & 0.7746 & 0.4000 & 0.4000 & 0.4899 \\ \hline & 1.0000 & 1.0000 & 0.4472 & 0 & 0 & 0 \\ & & 1.0000 & 0.4472 & 0 & 0 & 0 \\ & & & 1.0000 & 0 & 0 & 0 \\ \hline & & & & 1.0000 & 1.0000 & 0 \\ & & & & & 1.0000 & 0 \\ \hline & & & & & & 1.0000 \end{array}\right)$$

The entries C_{ij} reveal how closely terms i and j are related.