Information Retrieval Notes

Artem Vasenin

Last updated April 9, 2017

Contents

Chapter 1

Boolean Retrieval

1.1 Index

Any query is posed as a boolean expression of terms. First we create an index of words that occur in each text, this is done offline. E.g.

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello
Antony	1	1	0	0	0
Brutus	1	1	0	1	0
Caesar	1	1	0	1	1
Calpurnia	0	1	0	0	0
Cleopatra	1	0	0	0	0
mercy	1	0	1	1	1
worser	1	0	1	1	1

When we compute results of a query we first invert corresponding to NOT terms and then compute a bitwise AND and OR between vectors for each term. A few observations:

- The matrix is very sparse.
- Doesn't support more complex operations, such as proximity search.

1.2 Inverted Index

To make storing information more efficient we can use inverted matrix which only documents things that occur. The inverted index is a dictionary, with terms for keys and *posting lists* for values. A posting list is a sorted list which documents which documents the term occurs in.

Now to compute an AND query we compute and intersection of posting lists. This can be done in $O(n)$ where n is the number of documents in collection, in practice much faster.

When computing a conjunctive query with more than two terms we process them based on length of posting lists, in ascending order. For disjunctive terms we can estimate the size of result using the sum of sizes of disjuncts.