Text retrieval

Kristian Wichmann

February 9, 2017

1 Information retrieval vs. text retrieval

Information retrieval (IR) is the process of obtaining information from a source of relevant data. Very often, we will think of digital systems, and hence the amounts of data involved are very large. In general, we could be looking for any kind of information, like pictures, audio, or video.

Text retrieval (TR) is the subset of IR which deals with text. The most iconic example is web search based on a text query, like the services provided by search engines such as Google and Bing.

2 Push and pull modes of text access

Text retrieval happens in two major modes: push and pull.

2.1 Push mode

In push mode, the text is suggested (pushed) to the user based on previous interactions. This is what typically happens in a recommender system. For instance, Netflix will suggest movies and series based on your view history.

2.2 Pull mode

In pull mode, the user takes the initiative to get (pull) the text. This may be further subdivided:

- *Querying* - Here, the user knows what to look for, at enters a keyword text: a query. This works well when the user knows what he/she is looking for.

- *Browsing* - Here, the user navigates through the data, following a path enabled by its structure. This works well when the user wants to explore the data.

We will focus on querying in this document.

3 Terminology

Here we describe the various ingredients needed to make a *retrieval model*:

- We work with a *vocabulary* $V = \{w_1, \cdots, w_N\}$ of words.
- A *query* q may be written $q = (q_1, \cdots, q_m)$, where $q_i \in V$.
- A *document* can be written $d_i = (d_{i1}, \cdots, d_{im_i})$.
- A *collection* of documents is $C = \{d_1, \cdots, d_M\}$.

3.1 Term frequency-inverse document frequency

A given vocabulary word $w \in V$ can be described in terms of its appearence in a document/collection. Two common measures are *term frequency* (tf) and *inverse document frequency* (idf).

3.1.1 Term frequency

Term frequency directly describes the occurrence of w in d. There's many variations of this. Some of the simplest are:

- 0 or 1 depending on whether w is present or not.
- Raw frequency f_{wd}. A count of the number of occurences.
- Log normalization: $1 + \log f_{wd})$ when $f_{wd} \neq 0$, zero otherwise.

3.1.2 Inverse document frequency

The inverse document frequency describes how unusual a word w is in a collection C. If n_{wC} is the number of documents in C which contains w, the idf is defined as:

$$\text{idf} = \log \frac{N}{n_{wC}} \tag{3.1}$$

To avoid division by zero for non-occuring w, 1 is sometimes added to the denominator. A number of different weighting schemes for idf exists.

4 The text retrieval problem

Given a query q, we wish to extract the set of *relevant documents* $R(q) \subseteq C$ for the query.

In practice, all we can hope for is an approximation of the relevant documents: $R'(q)$.

5 Strategies

5.1 Document selection strategies

One way to solve the text retrieval system is to build a binary classifier f, which given a document d and a query q returns either 0 or 1, depending on whether or not $d \in R'(q)$:

$$R'(q) = \{d \in C | f(d, q) = 1\} \tag{5.1}$$

Of course, any way to choose $R'(q)$ is technically a binary classifier, but the idea is that f decides the *absolute relevance* of the document - there is no further nuance beyond "yes" or "no".

With this strategy, there's 2^N possible different outcomes, since each document is either relevant or not.

5.2 Document ranking

Instead, the function f might have a continuum of real values instead of just $\{0, 1\}$. Then we might choose $R'(q)$ based on a *cutoff* θ:

$$R'(q) = \{d \in C | f(d, q) > \theta\} \tag{5.2}$$

Here f is more nuanced, and decides what is called the *relative relevance* of the document. A list of documents sorted by decreasing relevance could be constructed, and θ decides where to stop the list. Or rather, if the user browses such a list, θ is decided by the user.

There's $N!$ different possible outcomes with this strategy, since each ordering of documents is distinct.

5.2.1 The probability ranking principle

The document ranking strategy is (under certain conditions) guaranteed by the *probability ranking principle* (PRP) to be of optimal utility to the user.

6 Vector space model

One approach to building a selection function is to use a *vector space model*. This is a *similarity-based* model, as it tries to give a measure of how similar the query and a document is.

6.1 The bag of words model

In the *bag of words* model (BOW), we disregard the ordering of the words in a document, and simply see them as an unordered list. Clearly, some information is lost in this simplification, but sometimes it will still yield useful results. The vector space model uses BOW.

6.2 Types of vectors

I each of the following cases, the relevant vector space for the model has a dimension of the size of the vocabulary V. In other words is has N dimensions.

6.2.1 Bit vectors

Here, each coordinate can only take on the values 0 or 1, indicating whether or not the word w_i is present in the query/document or not.

6.2.2 Frequency vector

Going a bit further than bit vectors, here coordinate i represents the number of times w_i appears in the document.

6.3 Similarity measure

No matter which model is used, the simplest similarity measure between a query and a document (or between two documents) is the dot product between the two:

$$\text{similarity} = q \cdot d \tag{6.1}$$

For bit vectors this is equal to the distinct number of query terms matched in the document. For frequency vectors, it is the total count of occurrences of any query term in the document.