Boolean Search with Positional Index

Eşref Özdemir

April 9, 2018

Contents

1 Data Preprocessing Steps

Data processing consists of the following steps:

- Parse sgm files and get texts inside body and title fields
- HTML sequence to ASCII conversion
- Tokenization by whitespace
- Punctuation handling
- Case-folding
- Stopword removal
- Stemming

1.1 Parsing

Each sgm file with 1000 documents were parsed and body and title fields of each document was stored. See parser.hpp documentation.

1.2 HTML to ASCII Conversion

Some of the documents contained various HTML sequences such as & to represent ampersand. Even when we remove the punctuation, having lots of amp tokens may affect the performance badly. For this reason, most frequenc HTML escape sequences were replaced by their equivalents or spaces. See doc_preprocessor.hpp documentation.

1.3 Tokenization

Splitting the documents to tokens was done by splitting the words by whitespace. See tokenizer.hpp documentation.

1.4 Punctuation Handling

Punctuation in the middle of the tokens may convey useful information. Additionally, removing these punctuation characters may transform different tokens into the same term (U.S. and us). For this reason, punctuation characters except .,<> were left in their places if they are in the middle.

As a second step, all non-alphanumeric characters at the beginning and at the end of a token were removed. See tokenizer.hpp documentation.

1.5 Case-folding

After the previously mentioned steps, all the uppercase letters were converted to their lowercase equivalents. See tokenizer.hpp documentation.

1.6 Stopword removal

All the stopword tokens were removed. See tokenizer.hpp documentation.

1.7 Stemming

Porter Stemmer was used to stem each word after the previous steps.

2 Statistics

During data preprocessing steps, the following statistics were collected:

- Total number of tokens in the corpus before normalization: 2719169
- Total number of tokens in the corpus after normalization: 2042869
- Total number of terms in the corpus before normalization: 129059
- Total number of terms in the corpus after normalization: 63860

Table 1: Top 20 most frequent terms before normalization

the	of	to	and	in	a	said	for	mln	The
on	it	is	said.	dlrs	from	that	its	vs	will

Table 2: Top 20 most frequent terms after normalization

to	said	mln	dlr	reuter	pct	from	vs	at	year
bank	compani	billion	ha	share	u.	ct	would	market	not

3 Positional Index

Positional index is constructed using hashmaps. For each normalized term, a vector from document IDs to a position vector is stored. C++ type of the positional index is as follows:

```
std::unordered_map<std::string,
                   std::vector<std::pair<size_t, std::vector<size_t>>>>
```

Dictionary written by indexer and used by searcher is a hashmap from strings to unsigned integers:

```
std::unordered_map<std::string, size_t>
```

Index written by indexer and used by searcher is a hashmap from term IDs to their document-wise position list:

```
std::unordered_map<size_t,
                   std::vector<std::pair<size_t, std::vector<size_t>>>>
```

4 Screenshots

4.1 Conjunctive Query

```
Please enter a search query and press Enter
> 1 price AND bbc
Matching documents:
18103
20039
```

4.2 Phrase Query

```
Please enter a search query and press Enter
> 2 spanish government
Matching documents:
18547
19786
```

4.3 Proximity Query

```
Please enter a search query and press Enter
> 3 citibank /3 reuter
Matching documents:
8633
19245
```