Personal Website, Low Level Design, Search Engine

Author: Yunfei Luo

May 17, 2020

---

1. Overview.

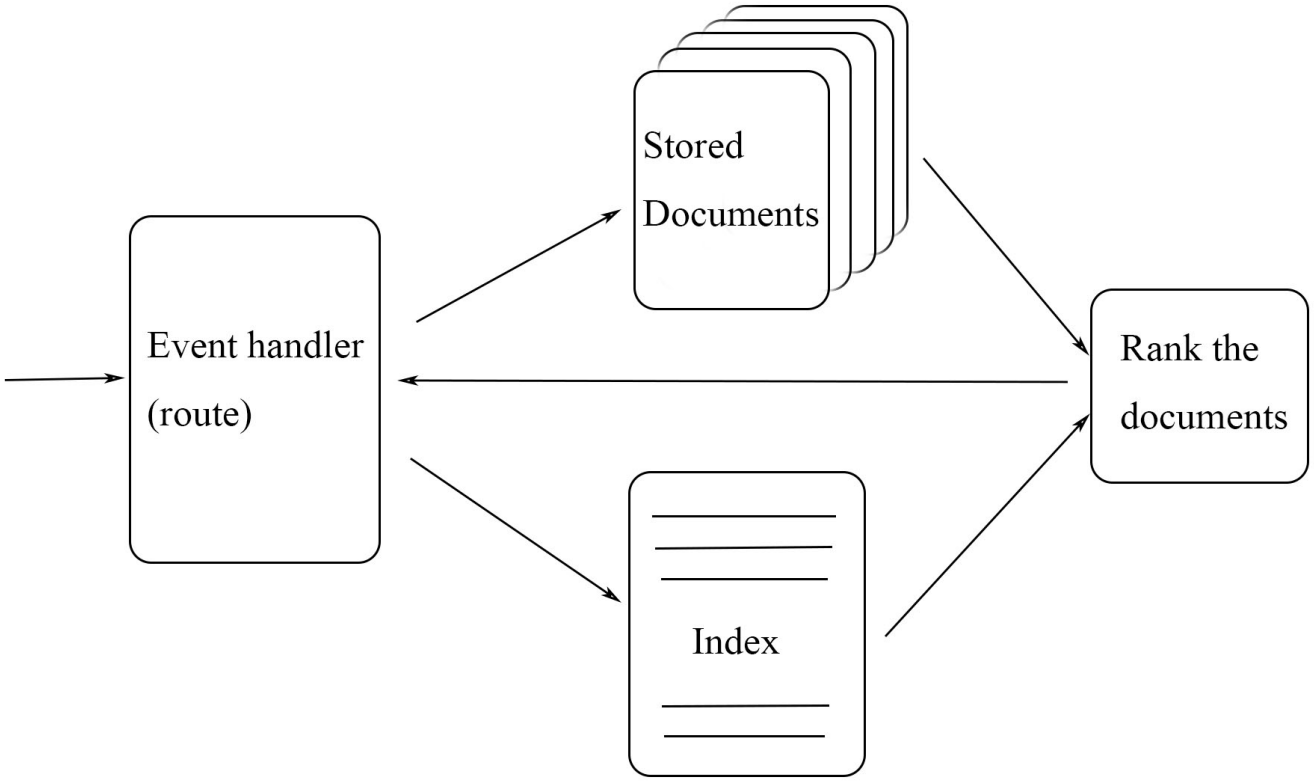

Figure 1. Micro Search Engine Architecture

2. Pre-processing.

2.1. Processing Documents.

This is independent with the interaction between frontend and backend. This process need to be done periodically, or whenever there are new articles added to the database.

The process follows steps:
i) tokenize, all letter to lower case, and remove dot between abbreviation words.
ii) stemming, use Snowball stemmer, also called Porter 2.
iii) stopword removal, remove non-significant words, use the concise list from python package: nltk.

2.2. Processing Queries.
When ever there is a query request coming from frontend, the query terms will first be pre-processed, then the querying functions will be executed. The processing steps are the same as steps for processing documents, see 2.1.

3. Indexing (Inverted Index).
Indexing will be mainly used by the retrieval model to rank the documents given the query terms. Inverted Index is a inverted list, that contains the necessary information for each stored terms. More specifically, the inverted index is defined as:

$$map : term \mapsto (map : document\_id \mapsto list(positions))$$

For example, term *learning* occur in document 1 and 3. It is the first and fifth word of document 1, and second and forth word of document 2, then the index for this term would looks like:

$$learning \mapsto \{document_1 \mapsto [1, 5], document_3 \mapsto [2, 4]\}$$

Then the term frequency in a document is $length(term.doc\_id)$, i.e. the total length of the list of positions in document with specified id. The collection frequency is the length of the list for a term.

4. Retrieval Models.
The key element for ranking the documents given the query terms. The following are the options of algorithms.

4.1. BM25. (formula information reference from: Search Engines, Information Retrieval in Practice, by W.B. Croft, D. Metzler, T. Strohman, 2015)
BM25 (BM stands for Best Match) is a well-known probabilistic retrieval model that not only take the document term frequency into account, but also consider the query term frequency. In the micro search engine, we will use the most common form of BM25, with no reference information. More specifically, the score for a document given the query terms is calculated by:

$$\sum_{i \in Q} \log(\frac{N - n_i + 0.5}{n_i + 0.5}) \cdot \frac{(k_1 + 1)f_i}{K + f_i} \cdot \frac{(k_2 + 1)qf_i}{k_2 + qf_i}$$

Where Q is the set of terms in the query terms, N is the total number of documents we have, n_i is the document frequency, i.e. number of documents that contain term i. f_i is the term frequency in the document, and qf_i is the term frequency in the query terms. The weighting parameters k_1, k_2, K are set empirically (by science). k_1 and k_2 determine the importance of document term frequency and query term frequency respectively. K is a normalized term, determined by:

$$K = k_1((1 - b) + b \cdot \frac{dl}{avdl})$$

Where b is a empirical parameter, dl is the document length, and $avdl$ is the average length of all the documents we have.

We could set the magic parameters to $k_1 = 1.1, k_2 = 10, b = 0.6$.

4.2. Query Likelihood. (formula information reference from: Search Engines, Information Retrieval in Practice, by W.B. Croft, D. Metzler, T. Strohman, 2015)
Query Likelihood is a well-known probabilistic retrieval model depends on language model. The

smoothing techniques we use is Dirichlet smoothing. More specifically, we calculate the score for a document by:

$$\alpha_D P(q_i|D) + \alpha_D P(q_i|C)$$

where $P(q_i|D)$ is the probability of query term i occur in document D, and $P(q_i|C)$ is the probability of query term i occur in the entire collection C. α_D is the Dirichlet smoothing coefficient determined by $\alpha_D = \frac{\mu}{|D|+\mu}$, where μ is set empirically. The final formula is:

$$\frac{f_{q_i,D} + \mu \frac{c_{q_i}}{|C|}}{|D| + \mu}$$

where $f_{q_i,D}$ is the term frequency in document D, and c_{q_i} is the term frequency in entire collection C. We could set $\mu = 1000$.