Information Retrieval – Query Completion based on breadth-first search in Wikipedia articles

Autoren / Verfasser: Peter Schögler, Fabian Henger
Abteilung/Institut für Interactive Systems and Data Science | *Institute of Interactive Systems and Data Science*

Problems & Objectives

The initial state of our situation is to find a way to auto suggest relevant results based on some input data e.g. keywords typed into an input field.

Therefore we are facing the following problems:

- Find a data source with a suitable public dataset
- Find a way to retrieve data from contemplable provider (API, Crawling, etc.)
- How to process and store the retrieved data
- How to find relevant results based on the search strings

And focus on some important objectives:

- Deliver a valuable dataset of source data
- Provide a fast query response
- Provide proper and suitable recommendations on input data

First Approaches and Analysis

The first steps we made included some fundamental analysis processes on existing data sources. Therefore we made some tests with the Europeana API Portal. Our conclusion was, that the API offers a lot of different very specific data which was not that easy to use for our aims.

Next we tried to find a way to use the Wikipedia database as source for further data processes and as a base for the query completion. There is a very big dataset which has to be preprocessed to get valuable data to work with.

We also analyzed the Wikipedia Web Search and the DOM structure of the Wikipedia articles.

Conclusion and results of the Analysis

According to our research and some tests we decided to retrieve the dataset on our own. Therefore we implemented a Web Crawler to retrieve data from specific Wikipedia articles with the goal to establish a structured and partly preprocessed own database.

Implementation Concept

The following Figure 1.1 shows a big picture of the principle concept of our implementation approach.

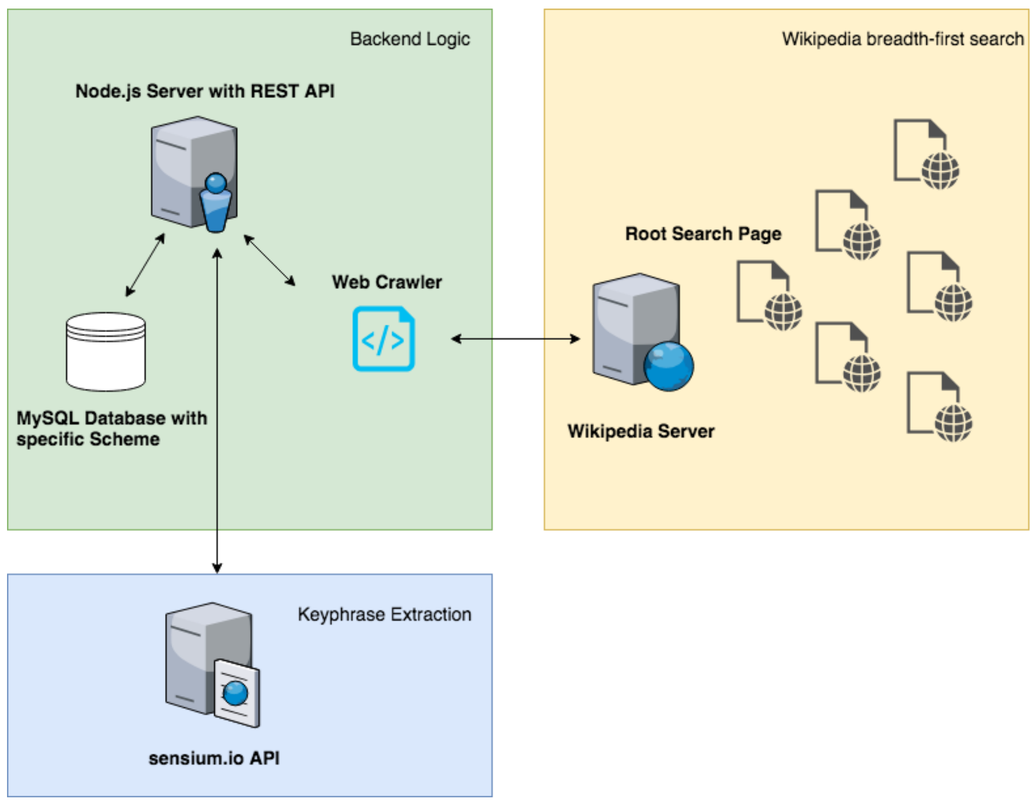

Figure 1.1 Big Picture of the implementation concept

Backend Logic – Web Crawler

We implemented a breadth-first search Wikipedia Web Crawler which starts with a base topic/URL and iterates through all links contained in that start URL. With that approach, we can focus on a specific start topic and crawl the content of linked topics of the parent article. This brings us some very useful advantages for the further data processing:

- Focus on specific topics on Wikipedia
- Include connected topics which are also relevant to the actual topic
- Minimize the divergence loss

The result data of the crawler is an array with elements of the following object structure:

- Link to the page
- Name of the link (important phrase)
- Content of the page (regarding to the DOM structure)

Backend Logic – Web Crawler

Our Backend node.js server delivers a REST API which implements one GET route to retrieve suggestions according to send a word or word phrase (string).

Backend Logic – Keyphrase Extraction

We use the API of sensium.io with the retrieved content from the crawler perform a keyphrase extraction to get the most important phrases out of the plain Wikipedia content text.

Backend Logic - Database

With the crawled data from Wikipedia and the keyphrase extraction of the content from the sensium.io API, we set up a scheme to perform a quite simple like search with column weighting. The weighting of the dataset is currently the following:

Feature	Weight
name (Wikipedia title of the url)	500
Keyphrase extraction key 10 - 1	Fibonacci numbers from 2 to 144
Content (if it contains search string)	1

We used up to ten keys of the keyphrase extraction and weighted them according to the Fibonacci row to increase the weighting according to the similarity of the search string and potential suggestions.

Evaluation

We evaluated our queried results by comparing the outputs of the Wikipedia Web Search and Google Search with our query completion output. Therefore we used the Topic “Real Madrid” as Root Crawling Page for our crawler and collected the content of the sub pages.

By entering the phrase “Jos” we expected the suggestion “José Mourinho” with the following result (including the Wikipedia link to the article):

Rank 1: José Antonio Camacho

Rank 2: José Mourinho

Rank 3: Josep Samitier

Rank 4: José María Zárraga

Rank 5: José Santamaría

These results are very plausible in the crawled context and deliver usable suggestions. Google for example delivers José Maurinho as first suggestion.