Final Report for Summer Course 2020

Chengrui Wang (2019201419)

July 20, 2020

Contents

1 Basic Information and Functions

The project is written totally in python 3, and the frontend part uses the python web framework Django.

The task of the course is to build a tiny search engine that is able to analyze, index, retrieve and mine the collected webpage pool. So of course this project has to support some functions including crawling webpages, analyzing information on them, and build up inverted index according to that.

This project can't deal with fuzzy searches, which is more complex and difficult, and optional to this course.

In specific, in this course, we are just dealing with all the webpages under the domain `http://info.ruc.edu.cn/`.

2 Usage

2.1 Prerequisites

To build and use this project, you should have the following software installed on your computer:

- `python 3.6+`
- `redis`

And also the following packages of python 3:

- `bs4` (for parsing HTML)
- `django`
- `jieba` (for dealing with content on the webpages)
- `redis`
- `zhon`

all the packages above are in the latest version.

2.2 Build

We have already created a script `build.sh` for building this project. So you can simply open a terminal in the project directory and execute it. If nothing wrong happens, you can try to access `http://localhost:8000` to search something you are interested in.

3 Implementation Details

3.1 The Crawler

We use many tricks to make the crawler as convenient as possible.

First, we use multi-threading while crawling the pages to accelerate the process. By default, the number of threads is 32.

Second, we use the file `local_test_data/todo_list.txt` to record the pages that are failed to fetch, and when you run the crawler for a second time, it will start from the urls in `local_test_data/todo_list.txt`.

Third, we use the database `redis` to store all the status of the urls (Success/In process/Failure). This is more reliable when the crawler is interrupted unexpectedly.

3.2 Index Builder

In order to process the content in the pages better, I downloaded a stopword list from the internet and created an extra dictionary manually.

There are nothing special besides that. I just implement this part according to the slides from class and wikipedia.

3.3 Search Engine

In general, we use tf-idf cosine scores to evaluate all the perspective results. There are also static weights attached to each page, which will also influence the ranking and the final result.

3.4 Frontend

The frontend part of this project is built up with Django, a python web framework. We also use Bootstrap 3 to prettify the overall appearance of the website.