UROP1100H Report

LI Yunqi

ylilo@connect.ust.hk

Abstract

The report is for HKUST Fall2022 UROP1100H Project ”Large-Scale Spatiotemporal Data Analytics and Learning”, which focuses on making a huge system to support many kinds of queries in this semester. Based on the map of Hong Kong SAR, China and the dataset from Hong Kong Statics Department, The system could show thousands of routes in real-life road network, or display huge dataset like COVID-19 spatiotemporal data and generates various charts based on Django and OpenStreetMap. Moreover, it could support user-defined dataset with specific format with file type *json* or *xlsx* etc. In this report, I will introduce my part of the system for displaying data and response to user-input which based on a open-source Github project[3].

1 Introduction

1.1 Project Architecture

The system is based on Django[1] and OpenStreetMap[2]. Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It does not seperate the frontend and backend of the system[1]. So, it could render the website easily with processed data. OpenStreetMap is a free wiki world map that supports kinds of API like Python module folium. It is releatively the most precise and convenient opensource map for spatiotemporal data visualization.

The project code is based on a Github project ”show-route-web-app”[3], I realize my data processing and website render parts based on its architecture.

1.2 Data processing

Based on Python, a powerful programming language for data analysis and visualization, we can easily get and process data with its module. We will firstly send request to database and get the raw datasets which contains tons of chaotic data data and each dataset’s formats are different. I will introduce a data preocessing scheme next section which shows a satisfied results on unformed data.

1.3 Website Render

Frontend render mainly based on a Python module folium and HTML5, which could directedly display data on OpenStreetMap. It could creat corresponding render functions and data structures in Python files and sends them to frontend HTML files. I will introduce the multi-website render scheme in Section 3 which could support seperating website by its functions.

2 A Data Processing Scheme

2.1 Raw Dataset from Hong Kong Statics Department

An important part for the system is getting the geographical data for 18 districts of Hong Kong. But for another dataset like population or COVID-19 dataset from Hong Kong Statics Department, their formats are different from geographical dataset. Not to mention that there are tons of chaotic data especially in COVID-19 dataset because of the decoding issue of Python.

I propose a data process scheme based on Python module Pandas and Geopandas, which could calibrate two datasets as one as long as those two raw datasets contain the specific labels for data. Now I take an example from displaying elderly population data to illustrate the scheme. Whcih the population data from Hong Kong Statics Department that contains lots of useless labels. Based on the scheme I could select useful labels of data and combine with the same labels geographical data of districts then display each district's static data of elderly.

Firstly we have two raw dataset V and W which V represents the displaying dataset, in this example is population dataset and W represents the geographical dataset. It will output a new dataset S which contains each district's geographical data and population data. The algorithm will generate a temporary dictionary called $Population$ which contains useful labels as keys. And we could generate a filiter function f to determine the qualified data in V from $Population$. Then it goes through the whole dataset V, selects data which satisfy f, and calculating the $adcode$ which is the most important for outputing dataset S. Actually we

Algorithm 1 Filter Data

Input: two datasets V and W
Output: dataset S

$T := Useful\ labels\ from\ V$
$Population := Dictionary\ from\ T$
$f := Filiter\ function\ from\ Population$
for each datagroup $v \in V$ **do**
    if $f(v)$ qualifies **then**
        Calculate adcode from v
        $Population$ add v with corresponding $keys$
    end if
end for
S = Concat $Population$ and W
return S

have three keys in $Population$: $adcode$, $population$, $elderly$, but we only need to calculate $adcode$ specifically. And the base code for $adcode$ is always 81000. Finally we return a new dataset to render functions, combining folium we can easily display the processed data.

2.2 User-defined Dataset

The system could support three user-defined data file types: *json*, *xlsx*, *csv*. We will always rewrite the user-input files to *json* file. And using class *JSONResponse* to define the input file status to ensure the input files has the correct types. Since the input data should follow specific format which user should identify eighteen districts' data seperatly, so we have no need to process it again. we could directedly send it to render functions to display.

2.3 Dealing With COVID data

Since COVID-19 data is a dataset contains COVID related buildings' geographical location and Chinese name. So, we use another seperate file to processed data with *name* rather than *adcode*. However, the algorithm is the same as previous.

However, at current step, I still need to download raw COVID19 dataset and process it, then send to the system to finally display it.

3 A Multi-website Render Scheme

This section will cover a multi-website render scheme to support different usage in one system with different render functions. Following the design principles of Django, We seperate the render functions in file "views.py", also we write seperate HTML file for seperate display tasks. And by creating a main entry of

the system, once the system starts, it will always direct user to main webpage, and the user could select functions.

And Django allows us to store the processed data at the backend so we have no need to consider the problem that the different websites need to transport data. Since all seperate render functions have full accsee to all processed data.

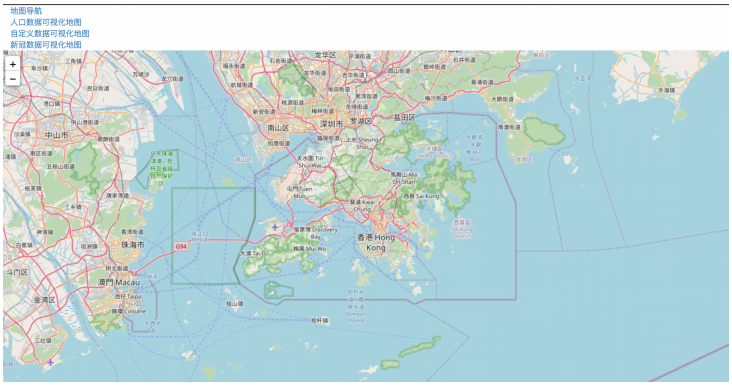

Figure 1: Screenshoot for main webpage

4 Demo Results

We have mainly four functions: displaying shortest route between two selected points; displaying population data on OpenStreetMap with charts; displaying user-input house data on OpenStreetMap with charts; displaying latest 7-days COVID-19 spatiotemporal data on OpenStreetMap.

Following serval figures show that screenshoot of different functions, some of them are taken once the finish writing the code. So, data for COVID-19 is not update.

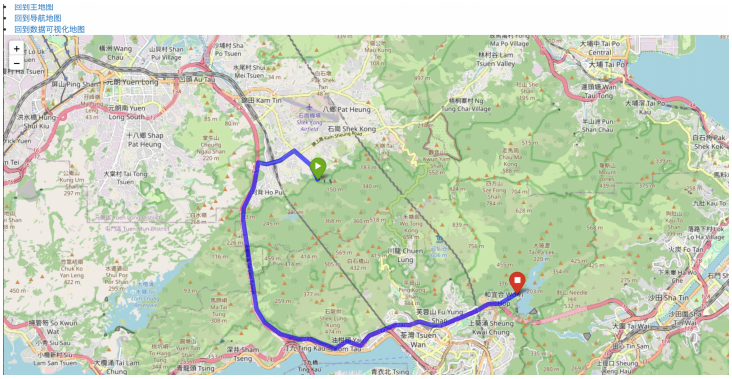

Figure 2: Screenshoot for show route between two selected points

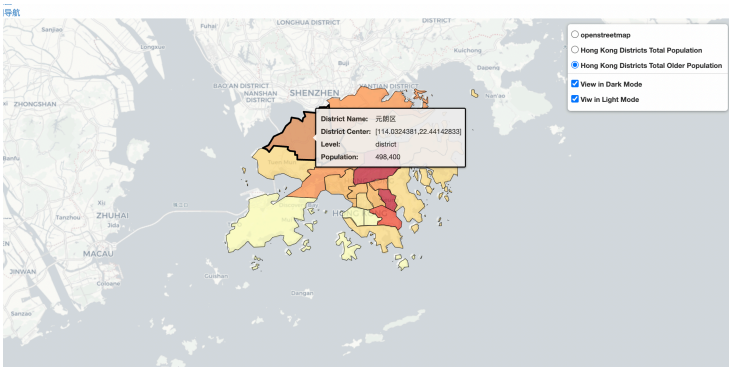

Figure 3: Screenshoot for displaying population data

Figure 4: Screenshoot for receving input

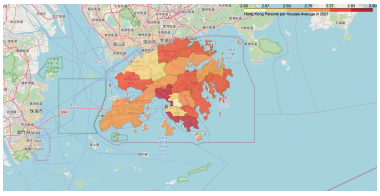

Figure 5: Screenshoot for displaying user-defined data

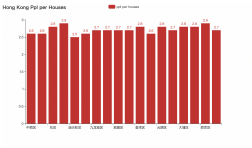

Figure 6: Screenshoot for chart1

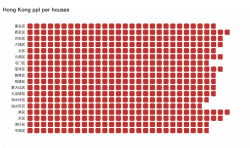

Figure 7: Screenshoot for chart2

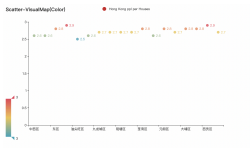

Figure 8: Screenshoot for chart3

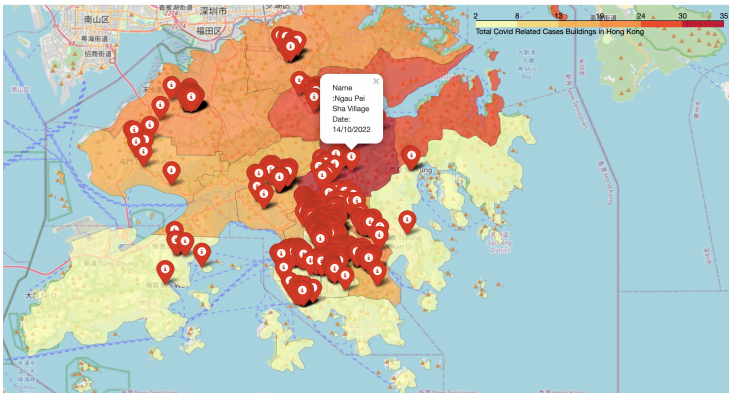

Figure 9: Screenshoot for displaying COVID19 data

5 Summary and Further Work

In this project I mainly work on processing data and displaying it on websites, I use Python modules to propose a data process scheme, and have learnt a lot of usage of website programming based on Django.

As now we could only receive three specific file types user input with pre-determined format, in the further we may need to propose new algorithm and data structure to receive arbitrary user inputs. Also, I will make the data processing for COVID19 relying on sending request to remote database rather than download-

ing and processing.

References

[1] Django. https://github.com/django/django.

[2] Openstreetmap. https://www.openstreetmap.org/#map=9/22.2446/114.9060.

[3] kumar7668. Show_map_route_webapp. https://github.com/kumar7668/Show_Map_Route_WebAPP.