UROP Report on Large-Scale Spatiotemporal Data Analytics and Learning

Name: YIN Zhuohao Student ID: 20677990 Email: zyinad@connect.ust.hk

Abstract

While conventional data science approaches have been applied on various domains to extract insights from large quantities of data, spatiotemporal data, due to its structural difference, have been poorly explored and exploited. While some classic machine learning algorithms are proven to work well with spatiotemporal data, such as K-means, K-Medoids, and EM algorithm, these methods can only handle limited data formats and lack interpretability. A remedy for low interpretability is through data visualizations.To this end, this UROP project seeks to develop a user-friendly API, which allows users to upload spatiotemporal data and produce interactive and informative visualizattions.

1. Overview

Spatiotemporal data refer to data that are related to both space and time. With the emergence of a variety of smart devices, such as mobile phones, smart watches, etc., an unprecedented amount of data have been recorded each day. However, studying these large-scale spatiotemporal data can be challenging and task specific and the insights are often presented without reference to the geospatial information visually. This project intends to implement a user-friendly visualization toolkit tailor-made for spatiotemporal data, where formatted spatiotemporal data can be directly fed into this API and produce interactive and intuitive visualizations for users to gain basic insights about how data are distributed over both space and time.

2. Related Work

2.1. Application Domains

Over the past decade, a plethora of learning algorithms have been performed on spatiotemporal for the purpose of extracting knowledge and utilizing the information. With proper methods, spatiotemporal data can be extremely insightful for multiple real-world problems in various domains. For example, in ecology, researchers would like to know the geospatial distribution of the habitats of different species and study the biological reason behind these facts (Roberts, 2010). In crime control, the locations where different type of crimes are committed are crucial information for both solving cases and preventing further crimes by properly allocate police forces to designated areas (Leipnik & Albert, 2002).

2.2. Classic Machine Learning Approaches on Spatiotemporal Data

K-means is a classic clustering algorithm that splits data into several possible clusters by evaluating the distance between each data point and the mean of each cluster. Being a straightforward and computationally simple algorithm, K-means has also been applied on spatial data. Sharma et al. (2012) performed the K-means algorithm on the Agricultural Statistics of India where they clustered the goegraphical locations according to rice production, using an interface called WEKA. However, one major drawback of their work is that the outputs are hardly interpretable by people without prior knowledge. Additionally, the outputs are not directly related to the geographical information, namely maps. Fig. 1 shows an example of their clustering outputs.

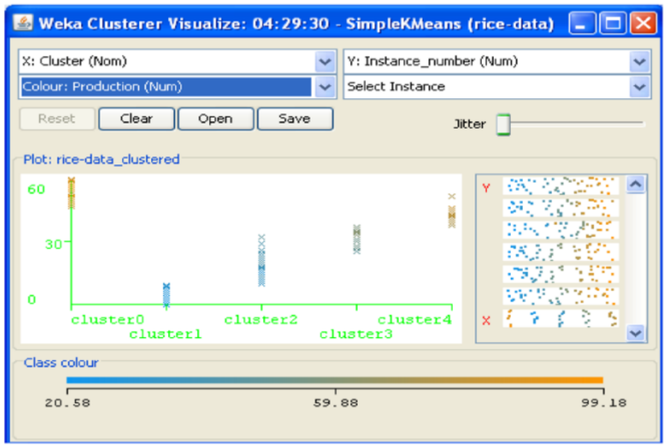


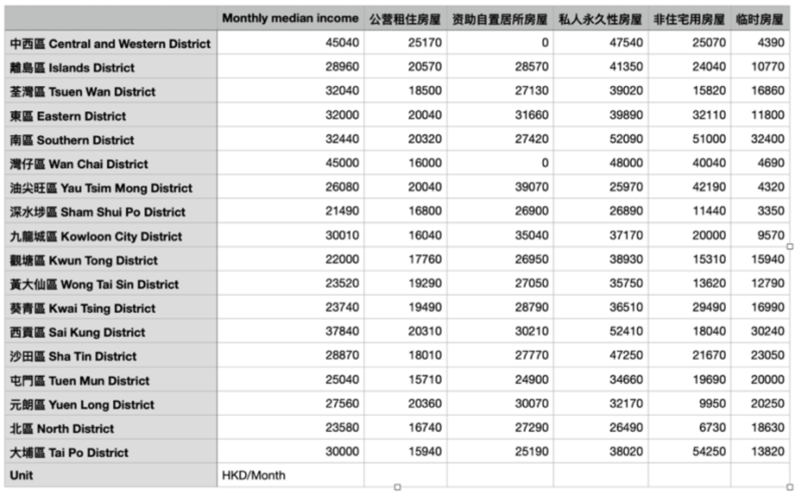
Figure 1. An example output of the K-means algorithm performed on spatial data.

2.3. Spatiotemporal Data Visualizations

Likeminded, Andrienko et al. (2003) proposed a range of techniques to visualize spatiotemporal data, including querying, map animation, focusing, linking and arranging views. Amongst these techniques, I found the focusing, linking and arranging views the most similar to my ideal outputs. Specifically, since spatiotemporal data are often in higher dimensions as opposes to other data, the concept is to present different subsets of data in different views according to user selections. My implementations are primarily based on this technique.

3. System Walkthrough

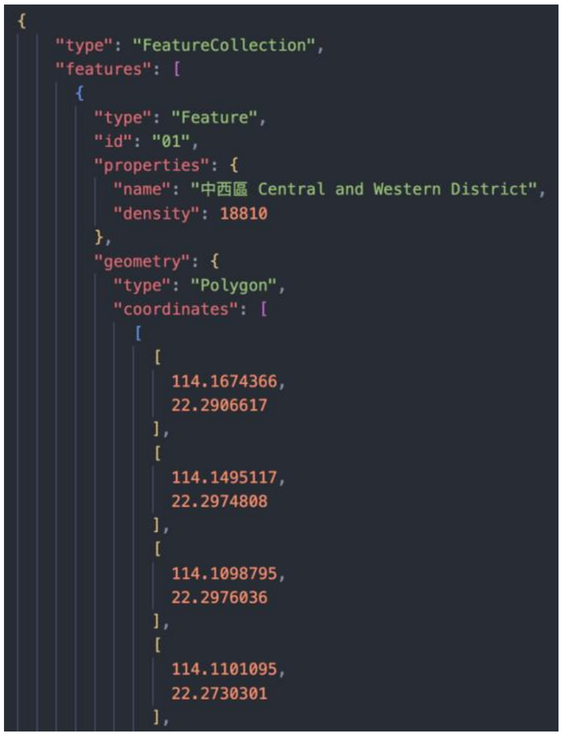
As this project serves as a prototype for possible future developments, the scope of spatial data is restricted within the city of Hong Kong only for demonstration purposes. The system is intend for interactive visualization of spatiotemporal data in Hong Kong. Therefore, each data file is associated with an administrative level and a short description. The administrative levels follow the same pattern as those on the OpenStreeMap Wiki. In specific for Hong Kong, an admin level of 6 separates the whole of Hong Kong into 18 administrative districts while an admin level of 5 regards Hong Kong as a whole. Data files must be configured in a way where each column stores some type of data for all 18 districts, with its semantic as the column name. Overall, the data files are assumed to be transformed into a Pandas DataFrame of shape [18, N], where N is the number of data for a single admin district. Fig. 2 shows a standard csv file under this realm.



	Monthly median income	公營租住房屋	资助自置居所房屋	私人永久性房屋	非住宅用房屋	临时房屋
中西區 Central and Western District	45040	25170	0	47540	25070	4390
離島區 Islands District	28960	20570	28570	41350	24040	10770
荃灣區 Tsuen Wan District	32040	18500	27130	39020	15820	16860
東區 Eastern District	32000	20040	31660	39890	32110	11800
南區 Southern District	32440	20320	27420	52090	51000	32400
灣仔區 Wan Chai District	45000	16000	0	48000	40040	4690
油尖旺區 Yau Tsim Mong District	26080	20040	39070	25970	42190	4320
深水埗區 Sham Shui Po District	21490	16800	26900	26890	11440	3350
九龍城區 Kowloon City District	30010	16040	35040	37170	20000	9570
觀塘區 Kwun Tong District	22000	17760	26950	38930	15310	15940
黃大仙區 Wong Tai Sin District	23520	19290	27050	35750	13620	12790
葵青區 Kwai Tsing District	23740	19490	28790	36510	29490	16990
西貢區 Sai Kung District	37840	20310	30210	52410	18040	30240
沙田區 Sha Tin District	28870	18010	27770	47250	21670	23050
屯門區 Tuen Mun District	25040	15710	24900	34660	19690	20000
元朗區 Yuen Long District	27560	20360	30070	32170	9950	20250
北區 North District	23580	16740	27290	26490	6730	18630
大埔區 Tai Po District	30000	15940	25190	38020	54250	13820
Unit	HKD/Month					

Figure 2. An example of the input data format of csv files.

A major bottleneck of spatial data visualization tools is that Javascript frameworks such as Leaflet takes geojson files as inputs while common people are unaware how to transform their data into geojson formats. It is manifest that there exists a gap between the common data formats (xlsx, csv) and the one that is recognizable by Javascript frameworks (geojson). Geojson files have a predefined format and follow the format strictly. Fig. 3 shows a segment of a geojson file.



```
{
    "type": "FeatureCollection",
    "features": [
      {
        "type": "Feature",
        "id": "01",
        "properties": {
          "name": "中西區 Central and Western District",
          "density": 18810
        },
        "geometry": {
          "type": "Polygon",
          "coordinates": [
            [
              [
                114.1674366,
                22.2906617
              ],
              [
                114.1495117,
                22.2974808
              ],
              [
                114.1098795,
                22.2976036
              ],
              [
                114.1101095,
                22.2730301
              ],
```

Figure 3. An example of geojson data format.

Thus, this project bridges the gap by taking csv files as inputs and preprocess the data to generate geojson files at runtime. This saves the users from figuring out the way to transform their own data and can benefit from the convenience by simply uploading their csv files and viewing visualization outputs.

This system handles single-data and multi-data scenarios differently. A single-data scenario is when there is only one data for an administrative area, such as population density, air quality index, etc., while multi-data scenario is when the number of data per area is larger than one. For single-data scenarios, the system yields a visualization on top of the map of Hong Kong, where each admin district is separated and filled by a color. The color is determined by the magnitude of the data inputted. Fig. 4 shows the visualized population density for each administrative district of Hong Kong.

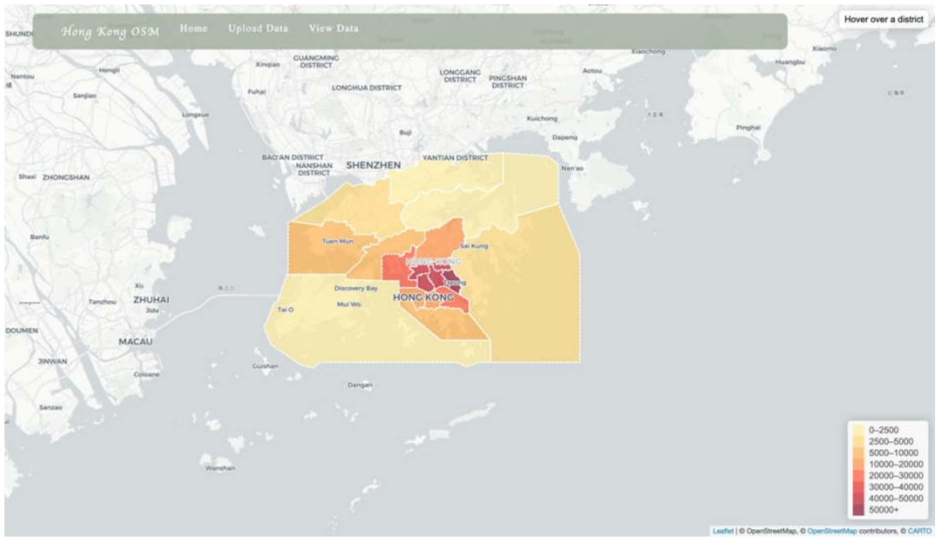


Figure 4. A visualization of population density of the 18 administrative districts in Hong Kong.

Under multi-data scenarios, each admin district will be associate with a bar plot displaying all columns in the input data. This type of plots are defined as horizontal comparisons. Apart from the horizontal comparisons, for each multi-data file, a vertical comparison will also be generated, where users can conveniently view a certain column of data for each admin district, as shown in Fig. 5.

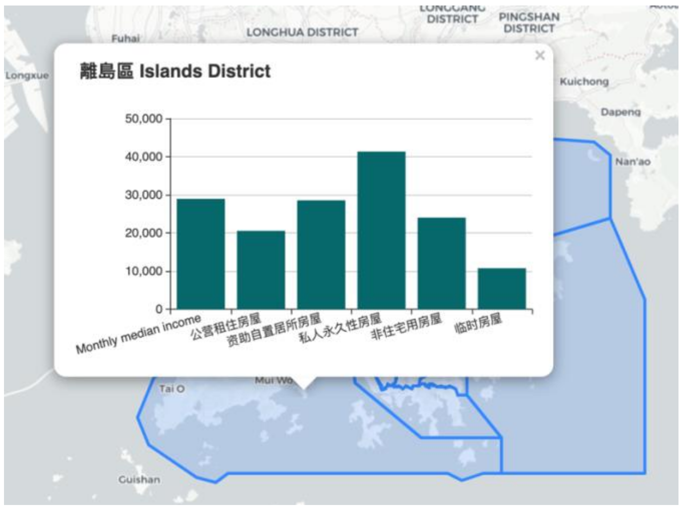


Figure 5. An illustration of horizontal comparison.

Furthermore, users may select two columns of data in the same file to simultaneously view these two columns and possible conlusions may be drawn in terms of correlations of the data. For more flexibility, users can also select the type of plot they desire out of ‘bar’, ‘line’, and ‘scatter’. An example can be seen in Fig. 6.

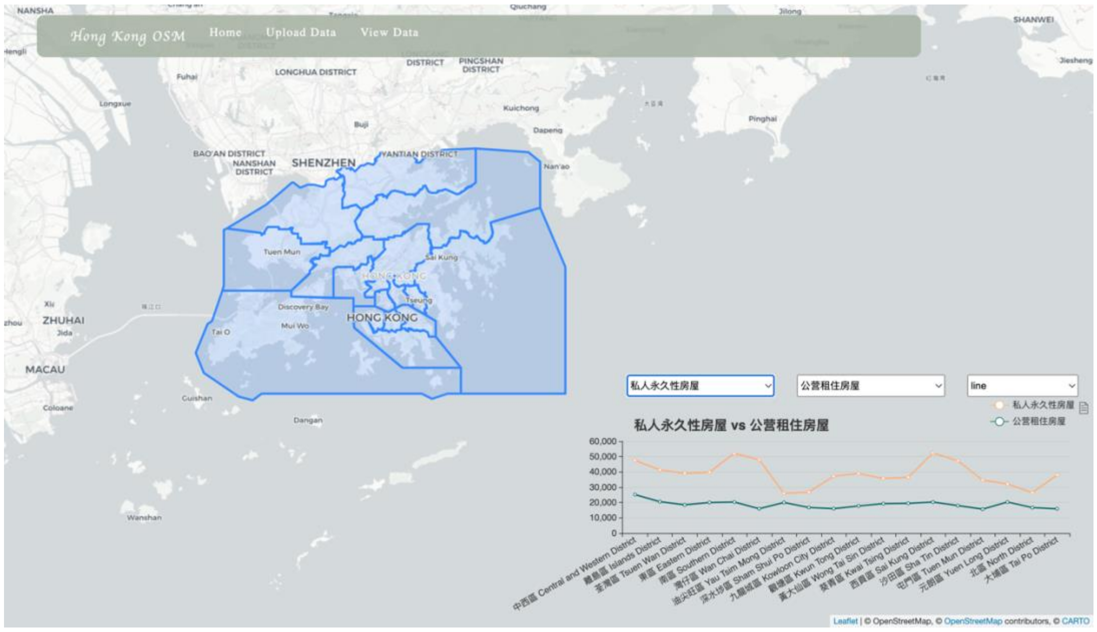

Hong Kong OSM
Home
Upload Data
View Data
私人永久性房屋
公营租住房屋
line
私人永久性房屋 vs 公营租住房屋
私人永久性房屋
公营租住房屋
60,000
50,000
40,000
30,000
20,000
10,000
0
中西區 Central and Western District
離島區 Islands District
荃灣區 Tsuen Wan District
東區 Eastern District
南區 Southern District
灣仔區 Wan Chai District
油尖旺區 Yau Tsim Mong District
深水埗區 Sham Shui Po District
九龍城區 Kowloon City District
觀塘區 Kwun Tong District
黃大仙區 Wong Tai Sin District
葵青區 Kwai Tsing District
西貢區 Sai Kung District
沙田區 Sha Tin District
屯門區 Tuen Mun District
元朗區 Yuen Long District
北區 North District
大埔區 Tai Po District
Leaflet | © OpenStreetMap, © OpenStreetMap contributors, © CARTO

Figure 6. An illustration of vertical comparison.

As can be seen in Fig. 6, the yellow line represents the income level for the people who live in private permanent houses while the green line represents that of the people who live in public rented houses. It is evident that the yellow line is lying on top of the green line, which indicates that people who live in private permanent houses earn significantly more money than those who live in public rented houses. Such insights can also be generalized to other types of data.

In short, the horizontal comparisons display all data for one given admin district while the vertical comparisons display a given column of data for all the admin districts.

4. Implementation Details

This project is primarily developed using the Django framework, together with several Javascript libraries like Leaflet and Echarts. This section will cover necessary details along the development process.

4.1. Backend

4.1.1. Models

Backend development in this project is done using the Django development framework using python. In models.py, I defined three models that stores data in the database. The first is AdminBoundary(), which is for storing the administrative boundaries of a city in forms of latitude and longitude, which are extracted using the overpass API. Each AdminBoundary() object stores a geojson file that is name by "city_admin_level.json". For instance, in this demo project, there is only one geojson file named "HK_6.json". The other two models are Json() and Table(), which stores data files in forms of json and csv respectively. In the Table() class, an admin level and a short description are also stored in the database.

4.1.2. Forms

As this system enables users to upload their own data, forms are used to receive user inputs in the frontend. There are two form classes, respectively GeoJsonForm() and TableForm() built on model Json() and Table().

4.1.3. Views

Views in Django are used to handle http responses and link to HTML templates. Each view is represented by a python function. There are four views in this project, respectively for the home page, the upload page, the overview of uploaded data and the data visualization page. One thing to notice is that data visualization pages are dynamically routed, meaning each uploaded data file will be displayed using a unique URL.

4.2. Frontend

Frontend components are essentially four HTML documents, responsible for the aforementioned four pages. The home page uses the Map.html as the template and all other templates extends Map.html. Details can be found in submitted source codes.

5. Discussion and Conclusion

In this project, I have explored different approaches to displaying spatiotemporal data and ahieved interactive visualization reults. However, there are limitations of the current system. For example, the system can be further developed to support swithcing between time stamps once large quantities of formatted data are available. Additionally, the mechanism of storing the data into the database can be further improved so that data entries in different files can be viewed and compared simultaneously.

References

Andrienko, N., Andrienko, G., & Gatalsky, P. (2003). Exploratory spatio-temporal visualization: an analytical review. Journal of Visual Languages & Computing, 14(6), 503-541.

Leipnik, M. R., & Albert, D. P. (Eds.). (2002). GIS in law enforcement: Implementation issues and case studies. CRC Press.

Roberts, J. J., Best, B. D., Dunn, D. C., Treml, E. A., & Halpin, P. N. (2010). Marine Geospatial Ecology Tools: An integrated framework for ecological geoprocessing with ArcGIS, Python, R, MATLAB, and C++. Environmental Modelling & Software, 25(10), 1197-1207.

Sharma, R., Alam, M. A., & Rani, A. (2012, August). K-means clustering in spatial data mining using weka interface. In International conference on advances in communication and computing technologies (ICACACT) (Vol. 26, p. 30).