OpenStreetMap Project - Data Wrangling with MongoDB

Author: Yuheng Cai

Map Area: Hongkong, China

References:

1. Mapzen.com to extract area map data: https://s3.amazonaws.com/metro-extracts.mapzen.com/hong-kong_china.osm.bz2 (https://s3.amazonaws.com/metro-extracts.mapzen.com/hong-kong_china.osm.bz2)
2. Characters in key: https://taginfo.openstreetmap.org/reports/characters_in_keys (https://taginfo.openstreetmap.org/reports/characters_in_keys)
3. Why doesn't Honghkong have postal code: https://www.quora.com/Why-doesnt-Hong-Kong-have-postal-codes (https://www.quora.com/Why-doesnt-Hong-Kong-have-postal-codes)

Source code: data_wrangling.py

1. Problems Encountered in the Map

The Hongkong OSM data was selected for the project given that I live around Hongkong area and more familiar with it.

Before the Hongkong OSM data was transformed to a predefined data model and imported to MongoDB, the data cleaning process was conducted, during which following problems were identifed:

- Street names are not standardized (some acronyms were used i.e. AVE, Rd, St)
- Upper case key used in 'tag' tag (not compliant with convention according to References[2]
- Both British and American English were used in 'color' key (i.e ‘colour/color')
- Name key consists of multiple forms (i.e. "name", "name:zh", "name:en", each of which refers to name in different languages)
- Phone format is not consistent (some with +country/area code; others without)

Note: postal code is seldom(if not never) used in Hongkong. Please refer to References[3] for some forum discussion. Previous review feedback/suggestion on doing some wrangling with postal code is not relevant for Hongkong OSM data.

Street Name Not Standardized

Python code (data_wrangling.py) converts all 'non-standard' street names based on following mapping before getting imported to MongoDB:
street_mapping = {
"St": "Street",
"AVE": "Avenue",
"Rd": "Road"
}

Upper case keys in tag

By convention(References[2]), characters in key should be in lower case. This issue was simply addressed by applying lower case funciton to the key during the parsing and data model transformation process in Python code.

Color key in both British and American English

Similar to the approach addressing the street name issue mentioned above, color keys were standardized to uniform 'color' during the parsing and data model transformation process in Python code.

Multiple forms for Name key

My interpretation for the Hongkong area OSM data having name keys in multiple forms such as "name", "name:zh", "name:en" is due to the fact that Chinese and English (potentially other languages as well) are well accepted in Hongkong. Instead of doing data cleaning, a better approach is to transform these data into a better model so as to capture this kind of information and put them under a dictionary such as below:
{"name":
{"default": "default name",
"en": "english name",
"zh": "chinese name"
}
}

Phone Number

Some 'nodes' in Hongkong area OSM data contain phone number in standard format with '+country/area' code; others don't. Cleaning was done by using regular express to find those without '+country/area' and add them into the transformed data model (update_phone function in data_wrangling.py).
For example:
34282828 => +852 3428 2828
2809 4426 => +852 2809 4426

2. Overview of the Data

This section contains basic statistics about the dataset and the MongoDB queries used to gather them.

File sizes

hong-kong_china.osm.............504 MB
hong-kong_china.osm.json.... 584 MB

Number of Documents

```
db.hongkong.find().count()
2681824
```

Number of Nodes

```
db.hongkong.find({'type':'node'}).count()
2445872
```

Number of Ways

```
db.hongkong.find({"type":"way"}).count()
233798
```

Number of Unique Users

```
db.hongkong.distinct("created.user").length
1471
```

Top 10 Contributing Users

```
db.hongkong.aggregate([{"group":{"_id":"created.user", "count":{"sum":1}}}, {"sort":
{"count":-1}}, {"$limit":10}])
{ "_id" : "hlaw", "count" : 509012 }
{ "_id" : "MarsmanRom", "count" : 236477 }
{ "_id" : "Popolon", "count" : 162804 }
{ "_id" : "Rebecca114", "count" : 121009 }
{ "_id" : "sn0wblind", "count" : 102497 }
{ "_id" : "fsxy", "count" : 100588 }
{ "_id" : "katpatuka", "count" : 97674 }
{ "_id" : "fdulezi", "count" : 80073 }
{ "_id" : "KX675", "count" : 77980 }
{ "_id" : "rainy3519446", "count" : 58229 }
```

Minority users contributes most data.

Number of Documents without Data Source

```
db.hongkong.find({"source": {"$exists": 0}}).count()
2667657
```

Top 5 Data Sources

```
db.hongkong.aggregate([{"group":{"_id":"source", "count":{"sum":1}}}, {"sort":
{"count":-1}}, {"$limit":5}])
{ "_id" : null, "count" : 2667657 }
{ "_id" : "bing", "count" : 4214 }
{ "_id" : "GPS", "count" : 3549 }
{ "_id" : "Bing", "count" : 1429 }
{ "_id" : "Yahoo hires", "count" : 1417 }
```

Majority of data source is unknown.

Top 10 appearing shop types

```
db.hongkong.aggregate([{"match":{"shop":{"exists":1}}}, {"group": {"_id":"shop",
"count":{"sum":1}}}, {"sort":{"count":-1}}, {"$limit":10}])
{ "_id" : "mall", "count" : 468 }
{ "_id" : "convenience", "count" : 447 }
{ "_id" : "supermarket", "count" : 317 }
{ "_id" : "car", "count" : 44 }
{ "_id" : "bakery", "count" : 42 }
{ "_id" : "books", "count" : 38 }
{ "_id" : "kiosk", "count" : 37 }
{ "_id" : "bicycle", "count" : 31 }
{ "_id" : "yes", "count" : 31 }
{ "_id" : "clothes", "count" : 28 }
```

3. Additional Ideas About the Datasets

Summary of atrributes of 'shop' related elements (attribute name appearing time >50)

In [55]:

```
import xml.etree.cElementTree as ET
from collections import defaultdict
import pprint
import re
import codecs
import json

# check if an element has 'tag' tag whose attributes contains 'shop' key
def contain_tag_elem_with_shop_attrib(elem):
    for sub in elem.iter("*"):
        if sub.tag == 'tag':
            if 'shop' in sub.attrib['k']:
                return True
    return False

# accumalate all shop related 'tag' attribute keys
shop_keys_dict = defaultdict(int)
for _, elem in ET.iterparse('hong-kong_china.osm', events=("start", )):
    if elem.tag == "node" or elem.tag == "way" :
        if contain_tag_elem_with_shop_attrib(elem):
            for sub in elem.iter("tag"):
                shop_keys_dict[sub.attrib['k']] += 1
    else:
        continue

# print out those relatively more significant 'tag' attribute keys
for key, count in shop_keys_dict.items():
    if count > 50:
        print "Key: {}, Counts:{}".format(key, count)
```

```
Key: shop, Counts:1869
Key: addr:housenumber, Counts:152
Key: addr:street, Counts:166
Key: name:en, Counts:605
Key: name:zh, Counts:517
Key: layer, Counts:141
Key: opening_hours, Counts:93
Key: name, Counts:1600
Key: website, Counts:90
Key: building:levels, Counts:191
Key: building, Counts:447
```

Suggestion for imporoving shopping related data

Hongkong is a well known great place for shopping. It attracts millions of international shoppers every year. It's strategically important to provide shoppers access to quality shopping related information.
One suggestion will be to provide mobile SDKs for developer to build all kinds of social/shopping apps by which a crowsourcing mechanism can be established to collect users generated shopping related content/data. To achieve this goal, two fundamental challenges need to be tackled:

1. Unique identifier for each data source to track data acqusition channel effectiveness

From above data overview, source of most data is unknown. Firstly, this makes it hard to identify the root problem of data quality issue when it happens. Secondly there is no way to track what data sources contribute more data. To facilitate crowsourcing, each data source has to be associated with an unique identifer so that data acquistion channels' effectiveness can be tracked and reviewed subsequently.

2. Richness of shopping related data

From above data exploration, very few shop related 'tag' element's keys have more than 50 appearances, which is an indicator of lack of richness for shopping related data. Hopefully by adopting crowsourcing approach, richness can be improved.

Conclusion

There were few problems encountered throughotu the project, most were data format issues related to standardising 'tag' keys. These issues were easier to address before importing data to MongoDB. To boost the usage and quality of Hongkong area OSM data, a crowsourcing approah was suggested. Based on characteristics of current OSM data for Hongkong area, two challenges being unique identifier and richness of shopping related info had been identified.