Wrangle OpenStreetMap Data

Oct. 18 2015

Ren Zhang, zhang_ren@bentley.edu

1 Problems Encountered in the Map

1. There are some incorrect street names and also some abbreviated street types.

2. There are a few incorrect zip codes as well.

I will talk about these two in order.

Street Names

After parse the raw xml data file and look at the street names, I find out that some street names are abbreviated, such as "Adams St", “Kelley Ct” and “Birmingham Pkwy”. I believe it is better if we updated these abbreviations with the actual words like “Street”, “Court” and “Parkway”.

I also noticed that some street names are incorrect, such as "Boylston Street, 5th Floor", which should simply be “Boylston Street”. Since I am pretty familiar with Boston, I also hard coded the corrections for these types of mistakes.

Zip Codes

The zip code of Boston area should starts with *02*. However, in the dataset, I noticed that there are a couple of zip codes not belonging to Boston area. For example we have a zip code start with *012*. It is not an invalid zip code but rather a zip code of location near Boston. Luckily these incorrect cases are very rare.

2 Data Overview

Here in this section I will provide some overview of the dataset.

Data file size

The original downloaded OpenStreetMap in XML format is of size 401 MB. I parsed it into JSON format with the street types corrected, and the resulting JSON file is of size 622MB.

The attached sample ".osm" file is a sample of the original XML file, with one of every fifty top level elements been selected.

Summary statistics of the dataset

Below, are some summary statistics of the dataset as well as the python code that are used to generate these results.

Number of documents: 2180736

```
collection.find().count()
```

Number of unique users: 1007

```
len(collection.group(["created.uid"], {}, {"count":0}, "function(o, p){p.count++}"))
```

Number of nodes: 1886049

```
collection.find({"type":"node"}).count()
```

Number of ways: 294201

```
collection.find({"type":"way"}).count()
```

3 Some more exploration on the data set

I also looked at the number of different ways data are generated in the data set.

Number of methods be used to create data entry: 26

```
pipeline = [{"$group":{"_id": "$created_by",
                       "count": {"$sum": 1}}}]
result = collection.aggregate(pipeline)
print(len(result['result']))
```

Proportion of contributions from top users

I looked at the users with most contributions to Boston area OpenStreetMap. The top three users are with their id and their proportions of the total contributions are as follows:

1. Crschmidt: 56.44%
2. jremillard-massgis: 20.12%
3. OceanVortex: 4.13%

Code for generating this result:

```
pipeline = [{"$group":{"_id": "$created.user",
                       "count": {"$sum": 1}}},
            {"$project": {"proportion": {"$divide" :["$count",collection.find().count()]}}},
            {"$sort": {"proportion": -1}},
            {"$limit": 3}]
result = collection.aggregate(pipeline)
result['result']
```

After a bit googling, I think I identified the top contributor on the web. Here is a link to his website.

Top ten cuisines in Boston area

I also looked at the top cuisines:

1. Pizza
2. American
3. Chinese
4. Italian
5. Mexican
6. Indian
7. Thai
8. Sandwich
9. Japanese
10. Asian

It is interesting to see that Pizza is the top one, I was expecting Lobster Rolls or the New England Fish and Chips to be on top, but they are not.

Code for getting this list:

```
pipeline = [{"$match":{"amenity":{"$exists":1}, "amenity":"restaurant", "cuisine":{"$exists":1}}},
        {"$group":{"_id":"$cuisine", "count":{"$sum":1}}},
        {"$sort":{"count":-1}},
        {"$limit":10}]
result = collection.aggregate(pipeline)
result['result']
```

Universities in Boston

Boston is famous for the many universities in the city, I also took a look at that as well. Just list a few here:

- Boston University
- Massachusetts Institute of Technology
- Suffolk University
- Harvard University

Code for generating the list:

```
pipeline = [{"$match":{"amenity":{"$exists":1}, "amenity": "university", "name":{"$exists":1}}},
        {"$group":{"_id":"$name", "count":{"$sum":1}}},
        {"$sort":{"count":-1}}]
result = collection.aggregate(pipeline)
result['result']
```

4 Other ideas about the datasets

Improving data quality by Cross-validating using USPS's API

One thought I have in mind related to improving the data quality is that, we might be able to use the DOTS Address Validation API from USPS to correct errors as well as add in more detailed address information into the dataset. Here is a link to the API service.

The benefit of doing so, is of course much better data quality with the address. One downside of this is that the API service is not free, we might want to ask our boss to see whether he will give us funds to do so.