

Contents

一 数据源.....	1
二 分析结果及及解决办法.....	1
1) 分析结果	1
2) 解决办法	2
三 数据抽取转换过程.....	2
四 数据统计结果.....	3
1 数据总数.....	3
2 Node 节点的数量.....	3
3 Way 节点的数量	3
4 Suppermarket 的数量	4
5 Highway 的种类以及个数.....	4
五 分析数据的问题.....	5
六 数据处理建议.....	6

一 数据源

选取的是上海地区的 OSM 文件，大小 54M。链接如下：

https://s3.amazonaws.com/metro-extracts.mapzen.com/shanghai_china.osm.bz2

二 分析结果及及解决办法

1) 分析结果

对原始的 OSM 数据进行分析后，发现如下一些问题

- 相同属性的取值不统一
 1. 当 tag 为 “way “时，k=oneway 对应的 v 值不统一，根据官方的解释，应该为 yes 或者 no 但在数据集中有的值为-1。需要统一为 yes 或者 no。官方解释链接如下：
<http://wiki.openstreetmap.org/wiki/Way>
 2. highway 和 Hwy 不一致，需要转换 Hwy 到 highway。

- 当 tag 为 node 或者 way 时，k=name 对应的 v 是中文名字时，有的 node 节点没有 k=name-en 属性。
- 当 tag 为 node 时，user 的值有些是中文。但绝大部分的 user 取值是英文字母。

2) 解决办法

根据以上列出的问题，相应的解决办法如下

- 属性取值不统一的，在抽取数据时，把不规范或者不统一的值修改成规范或者统一值。
- 增加 k=name-en 属性，其 v 的取值采用百度开放的翻译 API 接口翻译成英文。

代码片段如下：

```
def translate_func(text):
    httpClient = None
    myurl = '/api/trans/vip/translate'
    fromLang = 'zh'
    toLang = 'en'
    salt = random.randint(32768, 65536)
    sign = appid + text + str(salt) + secretKey
    m1 = md5.new()
    m1.update(sign)
    sign = m1.hexdigest()
    myurl = myurl + '?appid=' + appid + '&q=' + urllib.quote(
        text) + '&from=' + fromLang + '&to=' + toLang + '&salt=' + str(salt) + '&sign=' + sign

    try:
        httpClient = httplib.HTTPConnection('api.fanyi.baidu.com')
        httpClient.request('GET', myurl)
        response = httpClient.getresponse()
        result = eval(response.read())
        return result["trans_result"][0]["dst"]
    except Exception, e:
        print e
    finally:
        if httpClient:
            httpClient.close()
```

- 将 user 的取值为中文的，用汉语拼音方式改写。

代码片段如下：

```
if u'\u4e00' <= item.attrib["user"] <= u'\u9fff':
    node["user"] = lazy_pinyin(item.attrib["user"])
else:
    node["user"] = item.attrib["user"]
```

三 数据抽取转换过程

- 完成以下内容

- 1 只处理两种类型的顶级标记：“节点（node）”和“道路（way）”
- 2 **CREATED** 数组中的属性应该添加到键“created”下
- 3 经纬度属性应该添加到“pos”数组中，以用于地理空间索引编制。确保“pos”数组中的值是浮点型，不是字符串。
- 4 数据存为 JSON 格式并导入 Mongo 数据库
- 5 创建 Mongodb 数据库和集合

```
> use shaosm
switched to db shaosm
> show dbs
admin      0.000GB
examples   0.000GB
local      0.000GB
> db
shaosm
> db.createCollection<mapdata>
2017-09-22T21:20:28.988+0800 E QUERY [thread1] ReferenceError: mapdata is not defined :
@(<shell>):1:1
> db.createCollection("mapdata")
< "ok" : 1 >
```

四 数据统计结果

1 数据总数

```
> show dbs
admin      0.000GB
examples   0.000GB
local      0.000GB
shaosm     0.007GB
> use shaosm
switched to db shaosm
> db.mapdata.find().count()
83898
```

中文在 mongodb 里显示正常:

```
db.napdata.find({"name":"东方大郡东5"}).pretty()
{"_id": "ObjectID('95c51112e9e288040e27a2d1")", "name": "Eastern day county east 5", "node_refs": [ "936510597", "936510598", "936510599", "936510600", "936510601", "936510602", "936510603", "936510604", "936510605", "936510606", "936510607", "936510608", "936510609", "936510610", "936510611", "936510612", "936510613", "936510614", "936510615", "936510616", "936510617", "936510618", "936510619", "936510620", "936510621", "936510622", "936510623", "936510624", "936510625", "936510626", "936510627", "936510628", "936510629", "936510630", "936510631", "936510632", "936510633", "936510634", "936510635", "936510636", "936510637", "936510638", "936510639", "936510640", "936510641", "936510642", "936510643", "936510644", "936510645", "936510646", "936510647", "936510648", "936510649", "936510650", "936510651", "936510652", "936510653", "936510654", "936510655", "936510656", "936510657", "936510658", "936510659", "936510660", "936510661", "936510662", "936510663", "936510664", "936510665", "936510666", "936510667", "936510668", "936510669", "936510670", "936510671", "936510672", "936510673", "936510674", "936510675", "936510676", "936510677", "936510678", "936510679", "936510680", "936510681", "936510682", "936510683", "936510684", "936510685", "936510686", "936510687", "936510688", "936510689", "936510690", "936510691", "936510692", "936510693", "936510694", "936510695", "936510696", "936510697", "936510698", "936510699", "936510700", "936510701", "936510702", "936510703", "936510704", "936510705", "936510706", "936510707", "936510708", "936510709", "936510710", "936510711", "936510712", "936510713", "936510714", "936510715", "936510716", "936510717", "936510718", "936510719", "936510720", "936510721", "936510722", "936510723", "936510724", "936510725", "936510726", "936510727", "936510728", "936510729", "936510730", "936510731", "936510732", "936510733", "936510734", "936510735", "936510736", "936510737", "936510738", "936510739", "936510740", "936510741", "936510742", "936510743", "936510744", "936510745", "936510746", "936510747", "936510748", "936510749", "936510750", "936510751", "936510752", "936510753", "936510754", "936510755", "936510756", "936510757", "936510758", "936510759", "936510760", "936510761", "936510762", "936510763", "936510764", "936510765", "936510766", "936510767", "936510768", "936510769", "936510770", "936510771", "936510772", "936510773", "936510774", "936510775", "936510776", "936510777", "936510778", "936510779", "936510780", "936510781", "936510782", "936510783", "936510784", "936510785", "936510786", "936510787", "936510788", "936510789", "936510790", "936510791", "936510792", "936510793", "936510794", "936510795", "936510796", "936510797", "936510798", "936510799", "936510800", "936510801", "936510802", "936510803", "936510804", "936510805", "936510806", "936510807", "936510808", "936510809", "936510810", "936510811", "936510812", "936510813", "936510814", "936510815", "936510816", "936510817", "936510818", "936510819", "936510820", "936510821", "936510822", "936510823", "936510824", "936510825", "936510826", "936510827", "936510828", "936510829", "936510830", "936510831", "936510832", "936510833", "936510834", "936510835", "936510836", "936510837", "936510838", "936510839", "936510840", "936510841", "936510842", "936510843", "936510844", "936510845", "936510846", "936510847", "936510848", "936510849", "936510850", "936510851", "936510852", "936510853", "936510854", "936510855", "936510856", "936510857", "936510858", "936510859", "936510860", "936510861", "936510862", "936510863", "936510864", "936510865", "936510866", "936510867", "936510868", "936510869", "936510870", "936510871", "936510872", "936510873", "936510874", "936510875", "936510876", "936510877", "936510878", "936510879", "936510880", "936510881", "936510882", "936510883", "936510884", "936510885", "936510886", "936510887", "936510888", "936510889", "936510890", "936510891", "936510892", "936510893", "936510894", "936510895", "936510896", "936510897", "936510898", "936510899", "936510900", "936510901", "936510902", "936510903", "936510904", "936510905", "936510906", "936510907", "936510908", "936510909", "936510910", "936510911", "936510912", "936510913", "936510914", "936510915", "936510916", "936510917", "936510918", "936510919", "936510920", "936510921", "936510922", "936510923", "936510924", "936510925", "936510926", "936510927", "936510928", "936510929", "936510930", "936510931", "936510932", "936510933", "936510934", "936510935", "936510936", "936510937", "936510938", "936510939", "936510940", "936510941", "936510942", "936510943", "936510944", "936510945", "936510946", "936510947", "936510948", "936510949", "936510950", "936510951", "936510952", "936510953", "936510954", "936510955", "936510956", "936510957", "936510958", "936510959", "936510960", "936510961", "936510962",
```

```
> db.mapdata.find().count()
```

83898

2 Node 节点的数量

```
> db.mapdata.find({"type":"node"}).count()
```

74650

3 Way 节点的数量

```
> db.mapdata.find({"type":"way"}).count()
```

9248

4 Supermarket 的数量

```
> db.mapdata.find({"shop": "supermarket"}).count()
```

3

5 Highway 的种类以及个数

```
> db.mapdata.aggregate([{$group:{_id: "$highway",result: { $sum: 1 }}}])
```

```
{ "_id" : "services", "result" : 1 }
```

```
{ "_id" : "tertiary_link", "result" : 9 }
```

```
{ "_id" : "platform", "result" : 1 }
```

```
{ "_id" : "pedestrian", "result" : 21 }
```

```
{ "_id" : "track", "result" : 62 }
```

```
{ "_id" : "living_street", "result" : 10 }
```

```
{ "_id" : "steps", "result" : 21 }
```

```
{ "_id" : "path", "result" : 36 }
```

```
{ "_id" : "trunk_link", "result" : 64 }
```

```
{ "_id" : "service", "result" : 584 }
```

```
{ "_id" : "raceway", "result" : 2 }
```

```
{ "_id" : "construction", "result" : 25 }
```

```
{ "_id" : "cycleway", "result" : 33 }
```

```
{ "_id" : "motorway_link", "result" : 195 }
```

```
{ "_id" : "footway", "result" : 206 }
```

```
{ "_id" : "motorway", "result" : 247 }
```

```
{ "_id" : "primary_link", "result" : 75 }
```

```
{ "_id" : "trunk", "result" : 60 }
```

```
{ "_id" : "secondary", "result" : 454 }
```

```
{ "_id" : "secondary_link", "result" : 25 }
```

五 分析数据的问题

通过第一次数据分析，发现如下一些问题：

- 1 当 type 是 way（道路）的时候，相关一些属性在有的节点没有记录，比如是否 ‘railway’ 还是 ‘highway’ 等其他属性，并没有记录。这样就造成后面数据统计的时候数据不准确。比如：

.....

```
"user": "Steven Shen",
```

```
"type": "way",
```

```
"id": "240348083",
```

.....

- 2 当 type 是 way（道路）的时候，其 node_refs 的值里有些 node 其实并不存在。这些 node 并没有出现在 type 是 node（节点）。比如：

```
{
```

```
"node_refs": [
```

```
  "3911958650", 这个值在 type 是 node（节点）时并不存在。
```

```
  "3911958859"
```

```
],
```

```
"created": {
```

```
  "changeset": "45942077",
```

```
  "version": "2",
```

```
  "uid": "445671",
```

```
  "timestamp": "2017-02-09T09:37:50Z"
```

```
},
```

```
"user": "flierfy",
```

```
"railway": "rail",
```

```
"type": "way",
```

```
"id": "387966082"
```

```
}
```

六 数据处理建议

1 当 **type** 是 **way**（道路）时，有些节点属性缺失的情况，可能需要参考其他一些道路数据集，尽量将这些属性补全，如果没有，考虑将这些节点删除掉，因为这些节点对于后面的一些统计分析没有啥价值。

2 当 **type** 是 **way**（道路）时，对于 **node_refs** 的值里有些 **node** 其实并不存在的情况，同样需要参考其他一些数据集，或者放大采样范围，或许能将这一部分数据补齐。否则建议将这些不存在的 **node** 从 **node_refs** 里删除掉。