利用 MongoDB 整理 OpenStreetMap 数据

by: 吴启华
email: waynewqh@foxmail.com

选择区域：中国，广东省，广州市
https://www.openstreetmap.org/relation/3287346
https://mapzen.com/data/metro-extracts/metro/guangzhou_china/
选择原因：广州是个大城市，会有不少用户愿意贡献数据。而且对我来说，广州是我比较熟悉的城市，在进行清洗的时候会有一定帮助。

1. 地图存在的问题

在下载了广州的地图并制成了一个小样本，在数据清洗时应该关注 tag 中的 k 和 v 值。我认为应该分为两种情况，一是尽量统一 k 值即 keyword 的命名，二是对 v 值即 value 进行清洗。通过统计和观察样本中关键字，并且审核其在样本中对于的赋值，可以对此地图作出有针对性的清洗工作。（审核、清洗代码参照 audit_data.py）

在样本的统计审查中，大概有关键字总数 92%左右的关键字在所有关键字中出现的次数少于千分之一，而少于千分之一意味着在此地图中出现的次数可能少于 700 次，出现次数很少的关键字并不需要批量审核，如果有需要的话可以手动清洗。

邮编和电话号码等数字类型

如 postcode 和 phone 等赋值应该是数字（电话号码可能会有()+-等字符）的关键字，实际上在此地图中出现次数少于千分之一，完全可以直接手动清洗。数据审核的思路如下，比如邮编应该是 510 开头的 6 位数，而电话除了+86 或 020 开头，应该是 1 开头的 11 位数字或 8 位的数字，如果不符合这个规则的话就要逐条查看。

在样本数据中，有些电话号码中间插入了空格，或者用(020)代表区号或用 ‘-’ 分割区号，这些都需要改成纯数字会比较好，这样直接从数据库中调用就可以拨号。我的修改操作是把手机号码统一成不带+86 开头的 11 位数字，座机修改为 020 开头的 11 位数字。

实际操作中，通过审核邮编和电话的数据，我发现有惠州市、东莞市、佛山市、深圳市甚至是香港的地址混入了广州的地图中。因此在数据清洗中，利用邮编和电话等作为对照，找出非本市的地点也是一个可行的思路。

关键字统一性

关键字命名的统一性主要出在地点的名字上，虽然主要的关键字是’name’，但还有’name:zh’,’name:en’等代表不同语种的名字，在这里我的处理是在 shape_element 函数中把这些’name:’开头的字段放在’names’的次级字典内。

此外还有’official_name’,’alt_name’,’nat_name’,’old_name’,’short_name’等开头的关于名字的关键字，不过出现次数非常少，我认为忽略即可。

名字的规范化

通过审查地图数据，我发现地点的名字是重灾区。由于国内的习惯，地点的名字应该要是简体中文为主，而 name 应该是主要查询的关键字，因此应该重点审核。而关于名字的其他关键字主要是 names[‘zh’]和 names[‘en’]，其他则出现次数太少，不足以作为对照参考。

可以批量清洗的有问题对象分几种情况：

> name 的值是非中文
> name 的值是繁体中文
> 不少 name 的值是中文名后跟着一个英文名,即 name[‘zh’]+ ’ ’ +name[‘en’]
> name 缺失，但 names 内有其他名字

我的审核、清洗思路如下:
1.首先填补 name 的缺失，如果有 names[‘zh’]的话，则直接赋值给 name。
2.接着删除中文名后的英文名，我这里的方法是：

- 如果有 names[‘en’]且 names[‘en’]的字符串被包含在 name 里且字符串长度比 name 短，则从 name 中删去 names[‘en’]
- 如果有 names[‘zh’]且 names[‘zh’]的字符串被包含在 name 里且字符串长度比 name 短，则让 name[‘zh’]作为 name 的值

3.利用如 HanziConv 或 nstool 等库进行繁转简操作

地址类型

OpenStreetMap 的维基里对 Key:addr 的用法有一定的规范，除了某些地标和道路外，国内的市区内地址的一般格式是 xx 省 xx 市 xx 区 xx 路 x 号 x 栋 x 室，分别对应关键字 addr:province、addr:city、addr:district、addr:street 和 addr:housenumber。不过在此地图中，不仅关于地址方面的数据缺失很多，而且即使有次级分类也并非详尽。

地址的标准格式我认为是类似如{‘city’: ‘广州市’, ‘district’: ‘天河区’, ‘housenumber’: ‘116 号’, ‘postcode’: ‘510635’, ‘province’: ‘广东省’, ‘street’: ‘天寿路’}，从邮政上来说只要有 street 和 housenumber，基本可以定

位到 district 和 postcode 等其他信息，而 province 和 city 有必要的话也可以批量录入。

我的审核思路是，检查 province 是否广东省，city 是否广州市，以及 district 是否属于广州的次级区域，检查 street 的值最后一个字符是否’路’或者’街’等关键字，检查 housenumber 的最后一个字符是否数字或’号’、’栋’、’层’等关键字。

2. 数据概述

这部分是利用 MongoDB 的查询指令对数据集的初步统计

文件大小

guangzhou_china.osm …….. 434 MB
audited_guangzhou_china.osm.json …….. 505 MB

文档数量

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

2377064
```

节点的数量

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

2173325
```

道路的数量

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

203739
```

唯一贡献用户数量

```
> db.gz.distinct("created.user").length

1328
```

贡献前 10 多的用户

```
> db.gz.aggregate({"$group":{"_id":"$created.user","count":{"$sum":1}}},
                  {"$sort":{"count": -1}},
                  {"$limit":10})
{ "_id" : "snOwblind", "count" : 372444 }
{ "_id" : "羊角忠实黑", "count" : 265153 }
{ "_id" : "MarsmanRom", "count" : 244919 }
{ "_id" : "katpatuka", "count" : 162363 }
{ "_id" : "fsxy", "count" : 100372 }
{ "_id" : "fdulezi", "count" : 73310 }
{ "_id" : "jamesks", "count" : 63187 }
{ "_id" : "Austin Zhu", "count" : 53153 }
{ "_id" : "rainy3519446", "count" : 44190 }
{ "_id" : "单人旁", "count" : 43736 }
```

贡献次数为 1 的用户

```
>db.gz.aggregate({"$group":{"_id":"$created.user", "count":{"$sum":1}}},
                 {"$group":{"_id":"$count", "num_users":{"$sum":1}}},
                 {"$sort": {"_id": 1}},
                 {"$limit":1})
{"_id" : 1.0,"num_users" : 293.0}
```

3. 其他问题及改进建议

进一步审核数据的规范性

```
> db.gz.find({"name":{"$exists":1}}).count()

  64215

> db.gz.find({"address":{"$exists":1}}).count()

  1024
```

可以看出，这个数据集缺失的关键词条相当多，特别是在地址字段的数据。而对初步清洗工作也只进行了一部分，对数据集的清洗工作还任重道远。
比如：

```
> db.gz.aggregate({"$match":{"amenity":{"$exists":1}}},

                  {"$group":{"_id":"$amenity","count":{"$sum":1}}},

                  {"$sort":{"count": -1}},

                  {"$limit":10})

{ "_id" : "school", "count" : 816 }

{ "_id" : "parking", "count" : 750 }

{ "_id" : "restaurant", "count" : 668 }

{ "_id" : "bank", "count" : 327 }

{ "_id" : "bus_station", "count" : 299 }

{ "_id" : "toilets", "count" : 254 }

{ "_id" : "fuel", "count" : 252 }

{ "_id" : "hospital", "count" : 237 }

{ "_id" : "fast_food", "count" : 224 }

{ "_id" : "cafe", "count" : 150 }
```

关于设施字段的数据还算不少，但是……

```
> db.gz.aggregate({"$match":{"amenity":"fast_food"}},
                  {"$group":{"_id":"$name","count":{"$sum":1}}},
                  {"$sort":{"count": -1}},
                  {"$limit":10})
{ "_id" : "KFC", "count" : 48 }
{ "_id" : "McDonald's", "count" : 42 }
{ "_id" : null, "count" : 25 }
{ "_id" : "麦当劳", "count" : 25 }
{ "_id" : "McDonalds", "count" : 7 }
{ "_id" : "Subway", "count" : 7 }
{ "_id" : "肯德基", "count" : 7 }
{ "_id" : "真功夫", "count" : 6 }
{ "_id" : "Burger King", "count" : 5 }
{ "_id" : "麦当劳 McDonald's", "count" : 4 }
```

可以看到，代表麦当劳的名字就有”McDonald’s”、”麦当劳”、”McDonalds”和”麦当劳 McDonald’s”四种，这在对名字的初步审核中并没有审核出来，而这不统一的名字显然对查询是很不友好的，因此对名字进行进一步的规范化是接下来进行数据清洗的重点之一。

经纬度准确性问题

在用如 name 或 address 等字段进行审核时，发现了很多不属于广州市的节点，虽然通过交叉检验可以排除不少非本地数据，但个人找出一个方法统一处理。

```
> db.gz.find({'pos':{"$exists":0}, 'node':{"$exists":1}}).count()
  0
```

可以得知，所有的节点都有 pos 字段，而经纬度的准确性对于地图来说是非常重要的，不过其准确性需要和其他数据源对照。后续希望可以找到一个有效的方法，通过检查 pos 的数值是否落入广州的范围内，来排除不属于广州的节点。

检查列表

在 audit_data.py 的数据审核中，只涉及到初步的批量检查和修正，得出了不少需要手动检查的 check list。通过手工建立映射字典，可以入 case study 中的 exercise11 那样批量修正，虽然是一个繁重的工作，但很有必要。

改进建议

根据上面提出的问题以及审核中出现的问题，我作出了如下改进建议，但需要对实施改进后会带来的好处以及预期的问题进行思考，权衡利弊，决定是否收益会大于风险。

建议 1：统一规范化设施、地名的名字

好处：

1、该项改进有助于在数据库中的分类查询和统计。

2、方便与其他数据来源进行对照，填补缺失的数据。

预期的问题：

1、名字的规范化标准难以确定，如上述例子中究竟用”麦当劳”（统计出现 25 次）还是”McDonald’s”（统计出现 42 次）作为统一名字，每个人会有不一样的意见。

2、有些地名可能不存在公允的规范化名称。

建议 2：对在数据批量审核中得出的存疑数据列表进行手工修正

好处：

1、进一步提升数据集的质量。

预期的问题：

1、对地址修正的工作量较大，而且可能会在重新输入数据时又产生了新的问题。

2、考虑的问题不够充分，手工修正列表不够完备。

建议 3：删除非广州本地数据

好处：

1、提高数据集的准确性。

预期的问题：

1、如果利用数据各项的交叉对照来排除，可能不够充分。

2、如果利用经纬度来排除，若不借助外部接口（如百度地图的 API 接口），使用本地化算法可能会较复杂。如利用常规的空间多边形射线法，对于边数可能过万的地图疆界以及 230 万个数据的样本，计算量将需要百亿次以上。

建议 4：删除只包含经纬度及用户创建信息而无其他如名字、地址等有用信息的数据

好处：

1、大大降低数据集的臃肿度。

2、提升数据查询速度和准确性。

预期的问题：

1、存在破坏数据完整性的风险。

2、可能会因为删除数据较多而打击用户上传的积极性。

结论

在改进建议中，我认为建议 1 的好处大于风险，而建议 2 中手工修正即使会产生新的问题也至少会比原始数据的问题少很多，因此好处将大于风险。建议 3 我认为算法的复杂性带来的风险也很大，而且也不能保证外部数据（如广州边界的地理信息）的准确性，因此风险可能会大于收益，在此数据集中由于地点数据的缺失很多，因此提高准确性带来的收益并不高。对于建议 4，破坏数据完整性的风险难以评价，因此风险可能会大于收益。

通过对数据集的初步检查，可以发现这个广州的 OSM 数据非常不完善，而且数据的格式也并不规范，虽然针对关键字段的审核为数据集做出了一定的修正，但还远远不够，不过就项目而言也提供了一定的贡献和清洗思路。

本数据集中的数据可能大部分都是通过机器人按照经纬度（或许是利用 GPS）批量录入，而其他有用的关键字，如 name 只有 6 万多条、address 只有 1 千多条，相比总共 237 万条数据而言只是凤毛麟角，也就是还有很多需要补完的地方。希望 OSM 项目能够找到一个办法，鼓励数据贡献用户提供更多有用的数据。