产业结构与人口迁徙——新冠疫情潜在影响与次生灾害分析

曾　充 3180106183 计科 1803
杨宇昊 3160105521 金融 1601
张梁育 3180105674 电气 1802

2021 年 1 月 17 日

目录

1 设计需求

1.1 设计背景

新型冠状病毒肺炎（COVID-19，简称“新冠肺炎”）疫情肆虐全球多个国家，2020 年 3 月 11 日，世界卫生组织 (WHO) 正式宣布将新冠肺炎列为全球性大流行病。在全球抗击新型冠状病毒疫情的过程中，产生了前所未有的大规模疫情数据，利用大数据分析技术和方法能够协助发现病毒传染源、监测疫情发展、调配救援物资，从而更好地进行疫情防控工作。可视分析作为大数据分析的重要方法，将数据智能处理、视觉表征和交互分析有机地结合，使机器智能和人类智慧深度融合、优势互补，为疫情防控中的分析、指挥和决策提供有效依据和指南。

1.2 研究问题

新型冠状病毒肺炎对我国经济发展造成了巨大冲击，在疫情发展的不同阶段，影响到了人口流动的趋势，同时对中国的经济产业结构造成了一次巨大的冲击。因此，我们小组利用可视分析技术，充分关联多源数据，来展示疫情走势与人口流动的变化趋势以及中国经济产业结构转型的状态。

1.3 目标用户

1. 关注疫情影响的普通群众；
2. 需要分析及评估疫情影响并制定相关防控措施的政府部门；
3. 需要基于疫情影响做出合适决策，制定相关策略的企业与机构。

1.4 应用价值

借助全球疫情走势、各国患者数量变化对比，人口流动、患者年龄分布和各省披露患者的分布和全国各地能够救治新冠医院分布的变化趋势，有助于分析疫情传播模式、比较全球各地主要是我国各地的传播差异、检测异常传播事件，制定传播管控策略，再辅以经济产业结构转型的状态，能够更好地评估疫情对国民经济、企业生产等方方面面带来的影响，防控复工复产困难、物资供需失衡等疫情次生灾害的发生。

2 设计介绍

2.1 数据来源

新型冠状病毒肺炎疫情（简称新冠疫情）是全球重大突发公共卫生事件，不仅对我国医疗卫生体系提出重大挑战，也对我国经济社会造成了重大冲击。CSMAR 在疫情爆发后积极响应科研抗“疫”的理念，秉承为学术界提供一流研究数据的一贯精神，于国内率先推出了新冠疫情与经济研究数据库，助力学者开展经济、金融、管理等领域的新冠疫情相关研究工作。新冠疫情与经济研究数据库收录了疫情基本信息、人口流动、经济影响三部分数据。其中，疫情基本信息包括每日疫情数据、财政补助资金情况、确诊病例逗留地点分布、病患轨迹、医疗救治医院情况；人口流动包括省份及城市的迁入迁出人口比例；经济影响包括宏观经济、股市、上市公司的数据。 我们选取了 CSMAR 的

1. 中国新冠肺炎确诊病患活动轨迹表
2. 各地新冠肺炎医疗救治医院数量统计表
3. 各城市迁出人口比例表 (日)
4. 各城市迁入人口比例表 (日)
5. 分省份国民生产总值表 (年) 数据表
6. 国外新冠肺炎疫情动态表 (日)
7. 全球新冠肺炎确诊和死亡病例数统计表 (日)
8. 确诊病例小区分布表

其中，具体数据条目和主要字段如下所示：

数据名称	主要字段	条目数量
中国新冠肺炎确诊患者活动轨迹表	省份名称、城市名称、县/区名称，发布时间、病患编号、性别、年龄	7146
各地新冠肺炎医疗救治医院数量统计表	地区代码、地区名称、医疗救治医院数量	385
各城市迁出人口比例表（日）	统计日期、迁出地区、隶属省份、迁出目的地	2432164
各城市迁入人口比例表（日）	统计日期、迁入地区、隶属省份、迁入来源地	2432164
分省份国民生产总值表（年）数据表	年度标识、省份名称、第一产业生产总值、第二产业生产总值、第三产业生产总值	2089
国外新冠肺炎疫情动态表（日）	统计日期、国家名称、确诊病例、新增确诊、死亡总数、新增死亡	69932
确诊病例小区分布表	省份名称、城市名称、县/区名称、逗留地点、逗留人数	8067

2.2 数据预处理

我们综合使用 Python、Excel 等多种工具或者语言对数据进行清洗，以便得到可以放置在可视化作品当中的 JSON 数据格式。这里列举部分数据预处理过程。

2.2.1 医院数量分布数据

```
data = pd.read_excel("NCP_RegMedHosSta.xlsx")
data.drop(columns = "AreaCode",inplace=True)
data.columns = ['name','value']
data.drop(index=[0,1],inplace=True)
data['name'] = data['name'].str.rstrip('市')
data.to_json("hospital.json",orient="table",index=False)
```

2.2.2 迁入迁出人口数据

```
# 迁出数据
filenames = ["NCP_EmigrRatioD{}.xlsx".format(i) for i in ["","1","2","3","4"]]

data = dict()

for filename in filenames:
```

```
    data[filename] = pd.read_excel(filename)
    data[filename].columns =["统计日期","迁出地区","迁出地区代码","隶属省份","迁出目的地","迁出
                                                目的地代码","迁出人口比例"]
    data[filename].drop([0,1],inplace=True)
    data[filename].drop(columns=['迁出地区代码','隶属省份','迁出目的地代码'],inplace=True)
    data[filename]['统计日期'] = pd.to_datetime(data[filename]['统计日期'])
    data[filename]['统计日期'] = data[filename]['统计日期'].dt.date
    data[filename].sort_values(['统计日期','迁出地区','迁出人口比例'],ascending=[True,True,
                                                False],inplace=True)
    data[filename] = data[filename][~data[filename]['迁出目的地'].str.contains("省")]
    data[filename] = data[filename].groupby(['统计日期','迁出地区']).head(5)
    print(filename + " done")

result = pd.concat(data.values())
result.to_json("cleaned/emigrate.json",index=False)
print("succeed!")


# 迁入数据
filenames = ["NCP_ImmigrRatioD{}.xlsx".format(i) for i in ["","1","2","3","4","5"]]

data = dict()

for filename in filenames:
    data[filename] = pd.read_excel(filename)
    data[filename].columns =["统计日期","迁入地区","迁入地区代码","隶属省份","迁入来源地","迁入
                                                来源地代码","迁入来源地隶属省份","迁入人口
                                                比例"]
    data[filename].drop([0,1],inplace=True)
    data[filename].drop(columns=['迁入地区代码','隶属省份','迁入来源地代码',"迁入来源地隶属省份
                                                "],inplace=True)
    data[filename]['统计日期'] = pd.to_datetime(data[filename]['统计日期'])
    data[filename]['统计日期'] = data[filename]['统计日期'].dt.date
    data[filename].sort_values(['统计日期','迁入地区','迁入人口比例'],ascending=[True,True,
                                                False],inplace=True)
    data[filename] = data[filename][~data[filename]['迁入来源地'].str.contains("省")]
    data[filename] = data[filename].groupby(['统计日期','迁入地区']).head(5)
    print(filename + " done")

result = pd.concat(data.values())
result.to_excel("cleaned/immigrate.json",index=False)
result.reset_index(inplace=True)
result.to_json("cleaned/emigrate.json")
print("succeed!")
```

2.2.3 分省份 GDP 数据

```
data = pd.read_excel("NCP_ProvincialGDP.xlsx")
data.drop(index=[0,1,2],inplace=True)
data.columns = ['年度标识', '省份编码', '省份名称', '地区生产总值-第一产业', '地区生产总值-第二
                                                产业', '地区生产总值-第三产业']
data.drop(columns=['省份编码'],inplace=True)
data.columns = ['year','province','one','two','three']
data = data.reindex(columns=['one','two','three','province','year'])
data = data[data['province'] != '中国']
data.dropna(inplace=True)
data.sort_values(['year','province'],inplace=True)
data.to_json('provinceGDP2.json',orient='table',index=False)
file = open('provinceGDP2.json')
data = file.read()
c = eval(data)
years = [i for i in range(1952,2021)]
series_data = [[] for i in range(len(years))]
for i,year in enumerate(years):
    for j in c:
        j['year'] = int(j['year'])
        if j['year'] == year:
            series_data[i].append(list(j.values()))
d = open('test3.txt',mode='x',encoding='utf-8')
d.write(str(series_data))
```

2.2.4 患者年龄分布数据

```
data = pd.read_excel("NCP_CnfrmdActTrack.xlsx")
data.drop(index=[0,1,2],inplace=True)
data.columns = ['省份代码', '省份名称', '城市代码', '城市名称', '县/区代码', '县/区名称','发布
                                                时间','病患编号','病患','性别','年龄']
data.drop(columns = ['省份代码','省份名称','城市代码','城市名称','县/区代码','县/区名称','病患
                                                编号','病患','性别'],inplace=True)
data['发布时间'] = pd.to_datetime(data['发布时间'])
data = data.assign(年龄段 = pd.cut(data['年龄'], bins=[i for i in range(0,110,20)],labels = ['
                                                儿童','青壮年','中年','中老年','老年']))
data.drop(columns=['年龄'],inplace=True)
data['发布时间'] = data['发布时间'].dt.date
a = data.value_counts()
a = a.sort_index()
a.to_json('test.json',orient='table')
```

2.2.5 各地披露数据比例

```
data = pd.read_excel("NCP_CnfrmdCourtDisd.xlsx")
cleaned = []
cleaned.append(dict())
cleaned[0]['name'] = '中国'
cleaned[0]['children'] = []
cleaned[0]['value'] = 0

for index,row in data.iterrows():
    if row['省份名称'] is not None:
        cleaned[0]['value'] += row['逗留人数']
        for i in cleaned[0]['children']:
            if i['name'] == row['省份名称']:
                i['value'] += row['逗留人数']
                province = i
                break
        else:
            new_dict = dict()
            new_dict['name'] = row['省份名称']
            new_dict['value'] = row['逗留人数']
            new_dict['children'] = []
            cleaned[0]['children'].append(new_dict.copy())
            province = cleaned[0]['children'][-1]
    if row['城市名称'] is not None:
        for i in province['children']:
            if i['name'] == row['城市名称']:
                i['value'] += row['逗留人数']
                # city = i
                break
        else:
            new_dict = dict()
            new_dict['name'] = row['城市名称']
            new_dict['value'] = row['逗留人数']
            new_dict['children'] = []
            province['children'].append(new_dict.copy())
            city = province['children'][-1]
    if row['县/区名称'] is not None:
        for i in city['children']:
            if i['name'] == row['县/区名称']:
                i['value'] += row['逗留人数']
                xian = i
                break
        else:
            new_dict = dict()
            new_dict['name'] = row['县/区名称']
            new_dict['value'] = row['逗留人数']
            new_dict['children'] = []
```

```
            city['children'].append(new_dict.copy())
            xian = city['children'][-1]
    if row['逗留地点'] is not None:
        for i in xian['children']:
            if i['name'] == row['逗留地点']:
                i['value'] += row['逗留人数']
                break
        else:
            new_dict = dict()
            new_dict['name'] = row['逗留地点']
            new_dict['value'] = row['逗留人数']
            xian['children'].append(new_dict.copy())

with open('tmp.json',mode='x',encoding='utf-8') as f:
    f.write(str(cleaned))
```

2.3 设计框架

我们使用了如下的开发语言、开发框架和了开发工具。

1. Echarts
2. React
3. Python
4. Pandas
5. Node.js
6. VSCode
7. Pycharm
8. Excel
9. Git

2.4 模块规划

2.4.1 人口迁徙与医院数量分布

疫情爆发时期和我国的春运时间重合，对防控疫情的扩散造成了极大的挑战。在不同的时间点，不同城市的迁入和迁出人口不一样。同时，我国各个城市医疗设施完善程度不一样，在

图中标记处各个城市的医院数量，也可以探究迁徙方向和医疗设施完善的城市有无一定的相关性。

2.4.2 各省份产业结构变化趋势

我国各个省份之间发展不平衡，产业间比例也有所不同。疫情对我国经济上造成了很大冲击，不仅能体现在产业结构上，同时也体现在加速了省份之间的分化。我们采用散点图的方式，横轴为第一产业，纵轴为第二产业，而散点的点的颜色区分不同省份，点的大小为第三产业。可视化图表能够随着时间变化而变化。

2.4.3 患者随时间的年龄段比例

2.4.4 各个区域披露数据的比例

我国幅员辽阔，各个省份各个地区之间疫情态势差异巨大。为了展现清楚各个区域之间的疫情状态，我们采用旭日图的方式，最内层为省份，展现各个省份之间的疫情差异，依次精确到城市、区县和地区。旭日图（英文：Sunburst）是一种现代饼图，它超越传统的饼图和环图，能表达清晰的层级和归属关系，以父子层次结构来显示数据构成情况。旭日图中，离远点越近表示级别越高，相邻两层中，是内层包含外层的关系。在实际项目中使用旭日图，可以更细分溯源分析数据，真正了解数据的具体构成。

原型设计

2.4.5 世界疫情发展趋势

2.4.6 疫情最严重的国家排名

3 案例展示

3.1 主面板

3.2 人口迁徙与医院数量分布

3.3 患者随时间的年龄段比例

3.4 各省份产业结构变化趋势

我们可以看到在经济发展的重要转型时期，例如改革开放，整个图的格局会发生比较大的变化。从 2010 到 2020 年几年间，各个省份位于坐标轴上的位置相对固定。可以预见的是，以第一产业第二产业为核心的省份，收到经济冲击较大，而以北京、上海等发达城市以第三产业

为主，相对影响较小。同时，从客观的角度来讲，疫情加速了企业线上办公的进程，促进了产业的转型。由于 2020 年的完整 GDP 数据还没有公布，我们只能做一个初步预期，同时在统计局公布后（1/18）将数据集放进去可以和我们的猜想进行交叉验证。

3.5 各个区域披露数据的比例

借助颜色和形状的视觉通道，我们可以看到各个省份的披露患者数据的比例，为了更好地应对疫情的蔓延趋势，各个省份对于每一位病患所在区域进行严格把控。然而，疫情初期，很多对于疫情的追踪并不到位。从图中可以看出，尽快湖北是中国疫情最为严重的区域，但是其披露的数据仅仅只有全部披露数据的四分之一。这也印证了疫情初期的披露数据还有提升的空间。

3.6 世界疫情发展趋势

从图中可以看出，世界的疫情发展态势截止到目前为止依然是不容乐观，确诊人数长期处于上升状态。可以预计，在未来很长一段时间里，疫情都将是一个常态，成为生活当中尤其是有国外交流需求的生活当中的一个重要组成部分。

3.7 疫情最严重的国家排名

从变化的数据可以看出，尽管中国在早期是疫情最为严重的国家，但是很快就控制住了疫情。然而以美国为首的很多西方国家对于疫情没有足够的危机意识，即便在有中国的先例之后依然没有很好地控制住疫情，严重拖累了世界范围内疫情的诊断治疗，并且持续了非常长的一段时间，直至今天。