Stat 156 Final Project Part 1: Paper Summary

Ritvik Iyer and Samuel Gao

9/29/2021

Section 1: Research Question and Conclusion

In the paper *Machine Learning-Aided Causal Inference Framework for Environmental Data Analysis: A COVID-19 Case Study* [3], Kang et. al explore the causal effect of environmental factors on COVID-19 severity. Using a combination of "snapshot" socio-economic indicators alongside time-series observations for 166 Chinese cities over a 76-day period from January to April 2020, the authors construct structural causal models (SCMs) to estimate causal relationships. After performing a series of robustness checks, the authors find that the vast majority (89 out of 90) of factor-effect relationships have no causal effect on COVID-19 severity. This paper provides two main conclusions: First, the specified environmental factors were unlikely to worsen the COVID-19 pandemic of 2020 and second, a combination of machine learning methods for dimensionality reduction and feature selection alongside structural causal models provide more robust conclusions than previous methods when investigating causation in the context of observational data.

Section 2: Data Description

The data used for the analysis of the paper includes macroeconomic and demographic data on 166 cities in China, their travel volumes from Wuhan, and their environmental data time series. Macroeconomic and demographic data were extracted from the annual statistical bulletins and/or statistical yearbooks published by the provincial and/or city governments from 2019. Travel volumes and degree of activeness of each city were extracted from Baidu's location-based service database while environmental and meteorological data were acquired through the China National Environmental Monitoring Center and the Application Programming Interface of the BINSTD data trading company. Confirmed cases data were extracted from China's Center for Disease Control and Prevention and local news reports, and the data shown is the three-day moving sum in the individual cities.

Section 3: Data Cleaning

For data cleaning, we obtained the datasets from the authors, compared them with the real data, and corrected mistakes. As the data were collected from hundreds of different sources, we decided that determining the accuracy of the features created by the authors was more important (and feasible) than re-constructing the data set from every source. As a result, we spent a larger amount of time on data quality checks. For instance, while calculating GDP, the total GDP should equal the combination of the primary, secondary and tertiary sectors. Tests were performed to examine if the above identity holds and mistakes were corrected to generate the proper numbers. We found a total of 4 typos that were significant enough to need correction. The numbers were compared with the official government statistics to correct the typos. In addition, certain columns of the summary statistics were expressed in different units and discrepancies were corrected. For example, the GDP data collected was quoted in RMB, while the summary statistics from the author used USD. Using an exchange rate from 2019 of 6.91 RMB to 1 USD, GDP figures were converted to the USD equivalency and summary statistics were reported using converted figures. Other data corrections involved proportion to percent conversions and per capita statistics being scaled correctly. Lastly, the data were parsed into three clusters to represent the relative sizes of the cities. The first cluster is "Megacities" which includes

some of the largest cities in China, while the second and third clusters are "Major Cities" and "Common Cities", respectively.

Cleaning the COVID data set has proven to be the most challenging, since the process to clean the data will continue even after this project deadline. The case numbers were crosschecked with a dataset from Harvard Dataverse, and discrepancies were noticed on the 3 day moving sums. For example, the data on the city Fuzhou utilized 3 day moving sums at certain spots, while it used 2 day moving sums at other spots. The data would need to be cleaned and made sure that it is consistent in order for the future analyses to be more accurate. Moving forward, we will clean the data from Harvard Dataverse and apply a consistent 3 day moving average.

Section 4: Summary Statistics and Explanations

We have placed the reproduced summary statistics figures in the Appendix. The summary statistics tables we have reproduced are originally from Table S1 and S3 in the supplemental information paper [2]. From Figure 1, we observe a large spread (e.g. SD, IQR) in many variables, including Population, City Area, and GDP. Looking closer at the quartiles and extreme values of these variables, we can infer that this population of cities is not homogeneous. For instance, the difference between 75th percentile GDP and the min GDP is far smaller than the difference between the max GDP and the 75th percentile. This suggests that there are clusters of cities which possess common characteristics, which we need to account for when performing causal inference. These differences by city type are summarized in Figure 3, where the authors group mega-cities together into Cluster 1, large cities into Cluster 2, and smaller cities into Cluster 3. In this table, we can see that larger versus smaller cities have interesting differences in socio-economic indicators, such as elderly as a percentage of population, population density, and Wuhan (known to be the city of first major outbreak of COVID-19) travelers per thousand population. In contrast to the socio-economic indicators, there is considerably less spread for each environmental factor in the time-series data in Figure 2. This may be because the data is from a 3-day moving average over each variable, which reduces variability by smoothing sudden spikes. Another important point to note is that the statistics in Figure 2 are computed over every city and date. Therefore, if we were to segment this data by date or city, it could potentially tell a different story.

Figure 1: Socio-economic "Snapshot" Summary Statistics

	Mean	SD	Min	25th %ile	Median	75th %ile	Max	IQR
Population (in thousands)	5624.67	4029.79	720.96	3176.92	4666.55	7181.67	31243.20	4004.75
GDP (Billions USD)	66.02	81.53	5.13	23.14	39.94	72.06	552.18	48.92
Primary sector (Billions USD)	3.54	2.53	0.17	1.93	3.13	4.65	22.45	2.72
Secondary sector (Billions USD)	25.80	26.94	1.89	10.20	16.21	29.99	151.89	19.79
Tertiary sector (Billions USD)	36.68	57.06	2.85	11.53	19.10	35.68	427.53	24.15
Elderly population %	19.51	4.51	4.92	17.13	19.69	22.48	32.20	5.35
Hospital Beds (per thousand people)	6.22	1.22	3.82	5.43	6.10	6.90	9.67	1.47
Registered doctors (per thousand)	2.81	0.76	1.32	2.29	2.73	3.14	5.76	0.85
Registered nurses (per thousand)	3.19	1.01	1.27	2.51	3.03	3.60	6.72	1.09
City area (in km^2)	11733.64	9080.77	1459.00	6339.50	10238.00	14288.50	82402.00	7949.00
Population Density (people per km^2)	652.19	694.77	24.31	314.20	543.98	725.53	6729.49	411.33
GDP per capita (Billions USD per km^2)	10.50	5.28	4.01	6.57	9.06	12.92	29.00	6.35
Primary sector % of GDP	8.42	5.09	0.09	4.12	8.12	11.55	23.08	7.43
Secondary sector % of GDP	41.33	7.58	16.16	36.66	41.46	46.31	60.00	9.65
Tertiary sector % of GDP	50.24	8.08	33.55	45.07	48.62	53.57	83.52	8.50
Average degree of activeness (0-8)	5.36	0.64	2.98	5.05	5.47	5.76	7.08	0.71
Wuhan travelers (thousands)	23.98	85.02	0.00	0.00	2.65	9.09	691.87	9.09
Wuhan travelers (per thousand pop.)	6.01	23.30	0.00	0.00	0.45	1.52	187.25	1.52

Figure 2: 3-Day Moving Average Factor Summary Statistics

	Mean	SD	Min	25th %ile	Median	75th %ile	Max	IQR
PM2.5 (μg/m3)	46.67	31.34	3.67	27.33	39.67	55.67	349.00	28.34
PM10 (μg/m3)	70.27	38.73	6.33	42.67	63.67	89.33	378.00	46.66
SO2 (μg/m3)	10.33	7.41	1.67	6.00	8.00	12.33	92.00	6.33
CO (mg/m3)	0.81	0.35	0.20	0.60	0.73	0.93	4.50	0.33
NO2 (μg/m3)	25.26	11.17	2.67	16.67	24.00	32.00	87.00	15.33
O3 (μg/m3)	83.82	22.06	5.00	69.00	83.33	97.67	166.67	28.67
Relative humidity (%)	71.23	18.20	8.00	60.33	74.67	85.33	100.00	25.00
Atmospheric pressure (hpa)	991.77	50.30	644.33	984.00	1011.00	1018.67	1035.33	34.67
Wind speed (m/s)	2.23	1.31	0.10	1.40	1.90	2.97	11.47	1.57
Average air temperature	8.98	6.34	-22.00	5.00	9.17	13.17	27.67	8.17
Degree of activeness	3.59	1.34	0.31	2.42	3.78	4.68	8.81	2.26
Morbidity rate	0.02	0.11	-0.00	-0.00	0.00	0.00	3.21	0.00
New confirmed cases	6.17	34.26	0.00	0.00	0.00	2.00	1021.00	2.00

Figure 3 : Mean Feature Value by City Cluster

	Cluster 1	Cluster 2	Cluster 3
Population (in thousands)	19019.47	6266.86	4620.88
GDP (Billions USD)	380.27	97.45	36.97
Primary sector (Billions USD)	5.90	2.92	3.61
Secondary sector (Billions USD)	117.50	39.16	15.92
Tertiary sector (Billions USD)	256.87	55.37	17.44
Elderly population %	17.46	16.54	20.62
Hospital Beds (per thousand people)	6.37	6.73	6.04
Registered doctors (per thousand)	3.49	3.57	2.51
Registered nurses (per thousand)	4.30	4.32	2.74
City area (in km^2)	19653.14	10440.70	11702.39
Population Density (people per km^2)	2386.85	868.45	477.46
GDP per capita (Billions USD per km^2)	147.60	105.22	57.13
Primary sector % of GDP	1.85	3.47	10.47
Secondary sector % of GDP	32.57	39.54	42.45
Tertiary sector % of GDP	65.58	56.99	47.07
Average degree of activeness (0-8)	4.87	4.84	5.57
Wuhan travelers (thousands)	31.69	7.49	29.07
Wuhan travelers (per thousand pop.)	1.59	1.07	7.93

References

[1] Kang, Q. (2021, April 19). Kangqiao-Ctrl/EnvCausal: A causal inference framework for environmental data analysis. GitHub. Retrieved from https://github.com/kangqiao-ctrl/EnvCausal.

[2] Kang, Q., Song, X., & Xin, X. (2021). (publication). (B. Chen, Y. Chen, X. Ye, & B. Zhang, Eds.)Supporting Information: Machine Learning–Aided Causal Inference Framework for Environmental Data Analysis: A COVID-19 Case Study(pp. S1–S25). St. John's, NL: Memorial University of Newfoundland.

[3] Kang, Q., Song, X., Xin, X., Chen, Y., & Ye, X. (2021). (publication). (B. Chen , Ed.)Machine Learning-Aided Causal Inference Framework for Environmental Data Analysis: A COVID-19 Case Study. American Chemical Society. Retrieved from https://pubs.acs.org/doi/10.1021/acs.est.1c02204?ref=pdf.

Code Appendix

```
knitr::opts_chunk$set(echo = TRUE)
dat = read.csv("~/Documents/UC BERKELEY/STATISTICS/STAT 156
               /Final Project/GitHub/EnvCausal-replication/snapshot_data.csv")
levels(dat$City) = c(levels(dat$City), "Xiamen", "Lhasa", "Urumqi", "Chongqing")
dat$City[9] = "Xiamen"
dat$City[14] = "Lhasa"
dat$City[18] = "Urumqi"
dat$City[21] = "Chongqing"
filter(dat, abs(dat$PRIM + dat$SEC + dat$TERT - dat$GDP) >1)
dat$SEC[dat$City == "Changchun"] = 2495.4
dat$GDP[dat$City == "Xiamen"] = 5995.04
dat$PRIM[dat$City == "Chongqing"] = 1551.42
dat$GDP[dat$City == "Baoding"] = 3558
dat$TERT[dat$City == "Huzhou"] = 1393.2
filter(dat, abs(dat$PRIM + dat$SEC + dat$TERT - dat$GDP) >1)
dat$POP = dat$POP*10
dat$GDP = dat$GDP/6.91/10
dat$PRIM = dat$PRIM/6.91/10
dat$SEC = dat$SEC/6.91/10
dat$TERT = dat$TERT/6.91/10
dat$GDPpc = dat$GDP/dat$POP*1000
dat$Prim. = dat$PRIM/dat$GDP
dat$Sec.= dat$SEC/dat$GDP
dat$Tert. = dat$TERT/dat$GDP
dat$POPDENS = dat$POP/dat$Area*1000
summary(dat)
envdat = read.csv("~/Documents/UC BERKELEY/STATISTICS/STAT 156/Final Project
/EnvCausal-main/data/time_series/
                  3_day_moving_average/df_m3.csv")
summary(envdat)
# Libraries for data processing and math
import pandas as pd
import numpy as np

# Library for file path manipulation
import os

# Library for Mandarin to English translation
import pinyin

# Library for simplifying function calls
from functools import partial

# This gets the file path of the data folder in EnvCausal-replication
root = os.path.dirname(os.getcwd())
data_dir = os.path.join(root, 'data')

# Set the file path for the cleaned snapshot data set
snapshot_path = os.path.join(data_dir, 'cleaned_snapshot_data.csv')

# Set the file path for the cleaned cluster data sets
```

```
cluster1_path = os.path.join(data_dir, 'cluster1_snapshot.csv')
cluster2_path = os.path.join(data_dir, 'cluster2_snapshot.csv')
cluster3_path = os.path.join(data_dir, 'cluster3_snapshot.csv')

# Set the file path for the 3-day moving average data set
time_series_3_day_dir = os.path.join(data_dir,
                                     'time_series', '3_day_moving_average')
time_series_path = os.path.join(time_series_3_day_dir, 'df_m3.csv')

# Read snapshots data set
snapshots = pd.read_csv(snapshot_path)

# Rename features for clarity
column_name_map = {
                    'POP':'Population (in thousands)',
                    'Area':'City area (in km^2)',
                    'POPDENS':'Population Density (people per km^2)',
                    'GDP':'GDP (Billions USD)',
                    'PRIM':'Primary sector (Billions USD)',
                    'SEC':'Secondary sector (Billions USD)',
                    'TERT':'Tertiary sector (Billions USD)',
                    'Prim.':'Primary sector % of GDP',
                    'Sec.':'Secondary sector % of GDP',
                    'Tert.':'Tertiary sector % of GDP',
                    'GDPpc':'GDP per capita (Billions USD per km^2)',
                    'X.60yr.':'Elderly population %',
                    'BED':'Hospital Beds (per thousand people)',
                    'DOC':'Registered doctors (per thousand)',
                    'NRS':'Registered nurses (per thousand)',
                    'TVLR':'Wuhan travellers (thousands)',
                    'TVLR.':'Wuhan travellers (per thousand pop.)',
                    'ACTV':'Average degree of activeness (0-8)'
                   }

# Drop unnecessary columns
snapshots_general = snapshots.drop(
    columns=[x for x in snapshots.columns if x not in column_name_map]).rename(
    column_name_map, axis=1)

# Compute summary statistics
first_quartile = partial(np.percentile, q=25, axis=0)
third_quartile = partial(np.percentile, q=75, axis=0)
summary_stats_snapshot  = snapshots_general.apply([np.mean, np.std, np.min,
                first_quartile,
                 np.median, third_quartile, np.max],
                 axis=0, result_type='broadcast').T.round(2)

# Clean up snapshot summary table
summary_stats_snapshot.columns = ['Mean',
                          'SD',
                          'Min',
                          '25th %ile',
                          'Median',
```

```
                          '75th %ile',
                          'Max']
summary_stats_snapshot['IQR'] = summary_stats_snapshot[
  '75th %ile'] - summary_stats_snapshot['25th %ile']

# Display summary table
summary_stats_snapshot

# Read time series data set
time_series = pd.read_csv(time_series_path)

# Read time series data set
time_series = pd.read_csv(time_series_path)

# Rename features for clarity
column_name_map_2 = {
                        'PM2.5':'PM2.5 (ug/m3)',
                        'PM10':'PM10 (ug/m3)',
                        'SO2':'SO2 (ug/m3)',
                        'CO':'CO (mg/m3)' ,
                        'NO2':'NO2 (ug/m3)',
                        'O3':'O3 (ug/m3)',
                        'HUM':'Relative humidity (%)',
                        'PRES':'Atmospheric pressure (hpa)',
                        'WSPD':'Wind speed (m/s)',
                        'TEMP':'Average air temperature',
                        'ACTV':'Degree of activeness',
                        'Case':'New confirmed cases',
                        'MORB%':'Morbidity rate'
                    }

# Define utility functions for Chinese to English (pinyin) character translation
def eng_translator(phrase_list):
    translation_map = {}
    for phrase in phrase_list:
        pinyin_phrase = pinyin.get(phrase, format="strip", delimiter=" ")
        pinyin_phrase = ''.join(pinyin_phrase.split(' '))
        pinyin_phrase = pinyin_phrase[0].upper() + pinyin_phrase[1:]
        translation_map[phrase] = pinyin_phrase
    return translation_map

def translate_column_to_eng(df, col):
    unique_phrases = np.unique(df[col]).tolist()
    translation_map = eng_translator(unique_phrases)
    return df[col].map(translation_map)

# time_series.columns.values[0] = 'Date'
# time_series.columns.values[1] = 'City'
# time_series['City'] = translate_column_to_eng(time_series, 'City')

# Drop unnecessary columns
time_series = time_series.drop(
    columns=[x for x in time_series.columns if x not in column_name_map_2]).rename(
```

```
    column_name_map_2, axis=1)

# Compute summary statistics
summary_stats_time_series = time_series.apply([np.mean, np.std, np.min, first_quartile,
                np.median, third_quartile, np.max],
                axis=0, result_type='broadcast').T.round(2)

# Clean up snapshot summary table
summary_stats_time_series.columns = ['Mean',
                        'SD',
                        'Min',
                        '25th %ile',
                        'Median',
                        '75th %ile',
                        'Max']

# Add IQR
summary_stats_time_series['IQR'] = summary_stats_time_series['75th %ile'] - summary_stats_time_series['


# Display summary table
summary_stats_time_series

# Read cluster snapshots data set
cluster1 = pd.read_csv(cluster1_path)
cluster2 = pd.read_csv(cluster2_path)
cluster3 = pd.read_csv(cluster3_path)

# Rename features for clarity
column_name_map = {
                   'POP':'Population (in thousands)',
                   'Area':'City area (in km^2)',
                   'POPDENS':'Population Density (people per km^2)',
                   'GDP':'GDP (Billions USD)',
                   'PRIM':'Primary sector (Billions USD)',
                   'SEC':'Secondary sector (Billions USD)',
                   'TERT':'Tertiary sector (Billions USD)',
                   'Prim.':'Primary sector % of GDP',
                   'Sec.':'Secondary sector % of GDP',
                   'Tert.':'Tertiary sector % of GDP',
                   'GDPpc':'GDP per capita (Billions USD per km^2)',
                   'X.60yr.':'Elderly population %',
                   'BED':'Hospital Beds (per thousand people)',
                   'DOC':'Registered doctors (per thousand)',
                   'NRS':'Registered nurses (per thousand)',
                   'TVLR':'Wuhan travellers (thousands)',
                   'TVLR.':'Wuhan travellers (per thousand pop.)',
                   'ACTV':'Average degree of activeness (0-8)'
                  }

# Drop unnecessary columns
cluster1_general = cluster1.drop(
    columns=[x for x in cluster1.columns if x not in column_name_map]).rename(
```

```
    column_name_map, axis=1)
cluster2_general = cluster2.drop(
    columns=[x for x in cluster2.columns if x not in column_name_map]).rename(
    column_name_map, axis=1)
cluster3_general = cluster3.drop(
    columns=[x for x in cluster3.columns if x not in column_name_map]).rename(
    column_name_map, axis=1)

# Compute cluster-wise feature averages
summary_stats_cluster1  = cluster1_general.apply([np.mean],
                axis=0, result_type='broadcast').T.round(2)
summary_stats_cluster2  = cluster2_general.apply([np.mean],
                axis=0, result_type='broadcast').T.round(2)
summary_stats_cluster3  = cluster3_general.apply([np.mean],
                axis=0, result_type='broadcast').T.round(2)

# Concatenate results together
mean_feature_by_cluster = pd.concat(
    [summary_stats_cluster1, summary_stats_cluster2, summary_stats_cluster3],
    axis=1)

# Clean up snapshot summary table
mean_feature_by_cluster.columns.values[0] = 'Cluster 1'
mean_feature_by_cluster.columns.values[1] = 'Cluster 2'
mean_feature_by_cluster.columns.values[2] = 'Cluster 3'

# Display summary table
mean_feature_by_cluster
```