PHOTO: DARCY PETERKA

ELEN 6889, Large-scale Streaming Processing

Project - Coronavirus Sub-hotspots Trend and News Fusion in United States

Professor Deepak S. Turaga

May 7th, 2020, presented By

Yanwen Jing (yj2556), Yanlin Liu (yl4238), Changxu Luo (cl3875), Weihan Chen (wc2681)

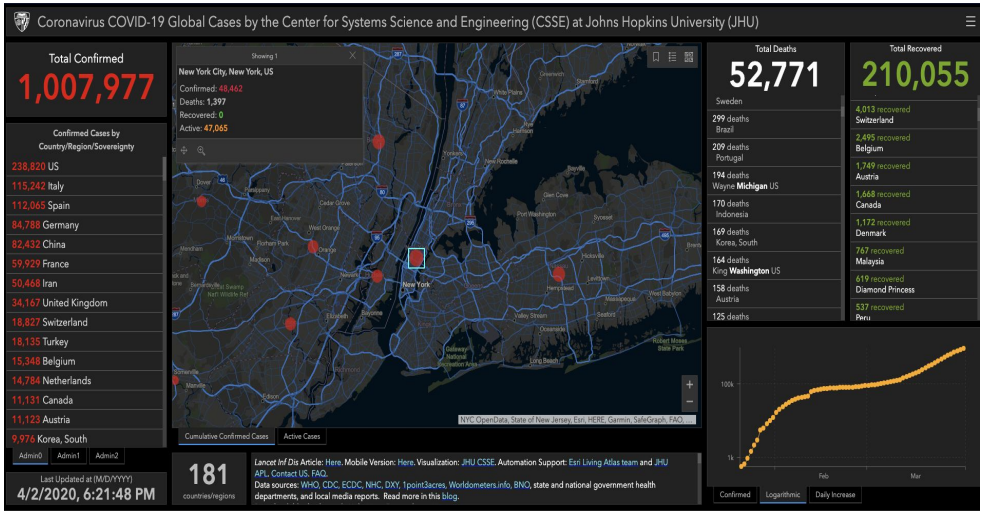
Problem

What we have now:

1. Coronavirus Cases Maps/Trending (https://coronavirus.jhu.edu/map.html)

2. Information from various News Organizations / US Government / Individual Users … (twitter)

The map has only the numbers for the area, but no corresponding information/news (government actions, recommendations, progress, etc.)



	14-Day Decline in Hospitalizations OR Under 15 new Hospitalizations (5-day avg)	14-Day Decline in Hospital Deaths OR Fewer than 5 deaths (5-day avg)	New Hospitalizations (Under 2 per 100K residents — 3 day rolling avg)	Share of total beds available (threshold of 30%)	Share of ICU beds available (threshold of 30%)	30 per 1k residents tested monthly (7-day average of new tests per day)	At least 30 contact tracers per 100K residents	Metrics Met
Capital Region	N	Y	0.58	41%	44%	N	325	4/7
Central New York	Y	Y	0.47	49%	51%	N	233	5/7
Finger Lakes	Y	Y	1.19	53%	64%	N	361	5/7
Long Island	Y	N	5.76	28%	26%	Y	852	2/7
Mid-Hudson	Y	N	4.74	31%	35%	Y	697	4/7
Mohawk Valley	Y	Y	1.17	58%	64%	N	146	5/7
New York City	Y	Y	5.41	26%	21%	Y	2520	3/7
North Country	Y	Y	0.08	53%	64%	N	126	5/7
Southern Tier	Y	Y	0.11	58%	52%	N	190	5/7
Western New York	N	Y	2.20	46%	40%	N	414	3/7

Problem

What we want now:

A US map with:

1. Daily updated coronavirus confirmed cases & deaths for each state
2. Plots about US confirmed cases & deaths trend over past few months
3. Corresponding information/news from various News Organizations/ US Government/ Individual Users … (on twitter)
4. Geographically distribution for the sub-hotspots and trending of them by a streaming application.

Solution

Data of cases statistics from John Hopkins University

Tags

#covid19
#covid
#coronavirus
#covid-19
...

US states location data

Deployer (Workflow Controller)

GCP Dataproc
(Stream Processing)

Cluster (New York)
- Twitter Streaming Session
- Spark Streaming Session

Cluster (California)
- Twitter Streaming Session
- Spark Streaming Session

Clusters (Other States...)

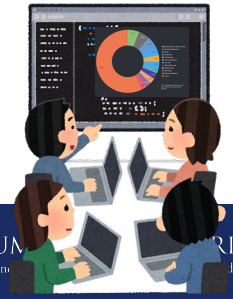
GCP BigQuery (Database)

GCP Storage (Temporary Storage)

Google Data Studio (Topics Visualization)

Front-end Data Server:
(For each state)
1. Cases Statistics
2. Trending Topics
3. IDs of tweets under trending topics

Django



Solution - confirmed cases & deaths

Dataset: on GCP BigQuery

COVID-19 Public Datasets

BigQuery Public Datasets Program

Datasets included in the COVID-19 public datasets program

VIEW DATASET

JOHNS HOPKINS UNIVERSITY

JHU Coronavirus COVID-19 Global Cases, by country

Johns Hopkins University

Repository of aggregated coronavirus COVID-19 cases by JHU

Solution - confirmed cases & deaths

Confirmed cases & deaths
By state & date

```
self.query_string2 = """
select state, date, sum(confirmed_cases) as confirmed_cases, sum(deaths) as deaths
from bigquery-public-data.covid19_usafacts.summary
group by state,date
"""

def get_cases_deaths(self):
    query_string = self.query_string2
    dataframe = (
        self.bqclient.query(query_string)
            .result()
            .to_dataframe()
    )
    self.cases_data = dataframe
```

Solution

```python
def sendData(c_socket, tags):
    """
    send data to socket
    """
    testLocation = [-74, 40, -73, 41]
    auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)
    twitter_stream = Stream(auth, TweetsListener(c_socket)) # bind t
    twitter_stream.filter(track=tags,
                          languages=['en'],
                          # filter_level='medium'
                          locations=testLocation
                          )


class twitter_client:
    def __init__(self, TCP_IP, TCP_PORT):
      self.s = s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
      self.s.bind((TCP_IP, TCP_PORT))

    def run_client(self, tags):
      try:
        self.s.listen(1)
        while True:
          print("Waiting for TCP connection...")
          conn, addr = self.s.accept()
          print("Connected... Starting getting tweets.")
          sendData(conn,tags)
          conn.close()
      except KeyboardInterrupt:
        exit
```

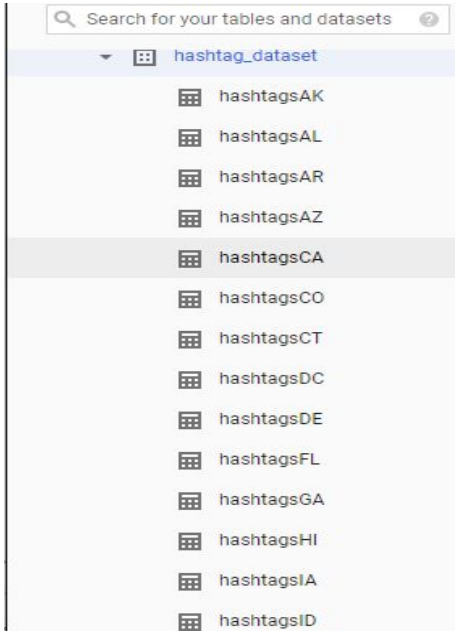
```
Waiting for TCP connection...
Connected... Starting getting tweets.
TEXT:RT @Thumbel28421928: Deathbed FaceTime calls
iPad visitors only
Nurses stifle sobs

#coronavirus #CoronaHaiku
#QuarantineHaiku #CovidHaiku…

TEXT:RT @Shem_Infinite: Don't forget, tomorrow at 12 PM EST
questions…
```

Twitter Streaming Session

Get the latest streaming data from twitter

Solution

```
-------------------------------
Time: 2020-05-01 05:08:25
-------------------------------
('#covid19', 98)
('#MayDay2020', 3)
('#lockdown', 3)
('#Odisha', 3)
('#India', 2)
('#bestgames', 2)
('#BREAKING:', 2)
('#Ohio', 2)
('#caceroleatelachota', 2)
('#HospitalPlaylist', 2)
...
```



Spark Streaming Session

Process the Data and Store them in Google Bigquery

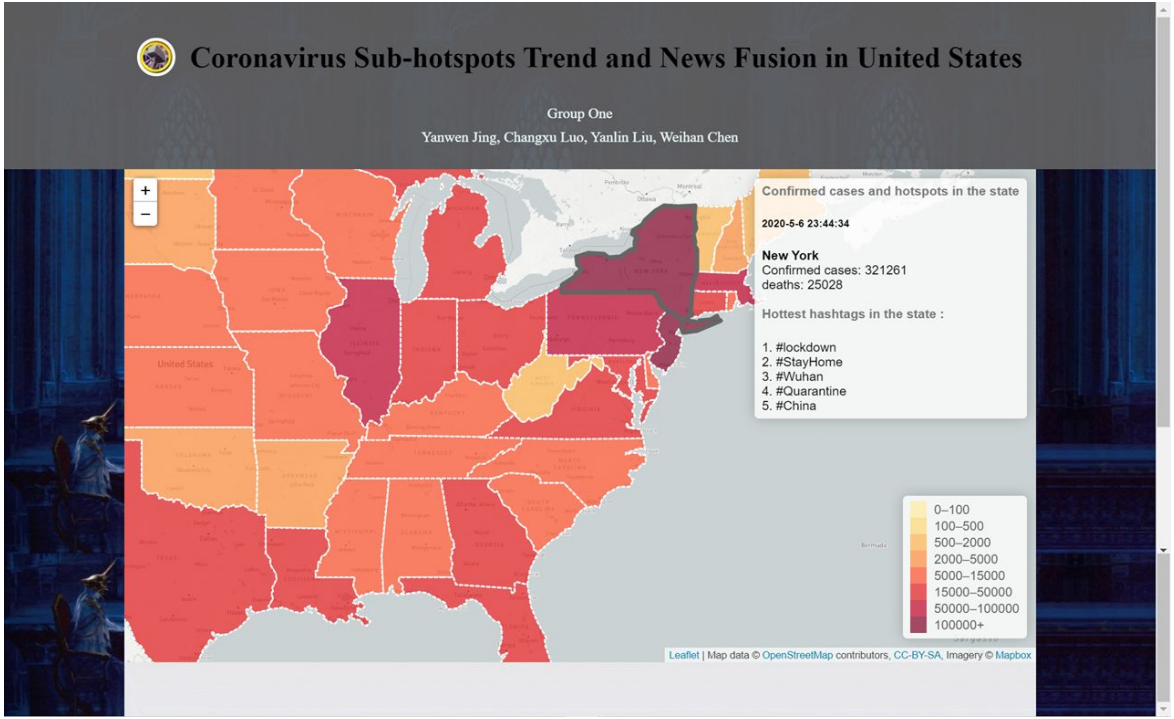
Solution

```
QUERY = (
    'SELECT * FROM `hashtag_dataset.' + '%s'%i +'` '
    'ORDER BY count DESC '
    'LIMIT 5')
query_job = client.query(QUERY)  # API request
rows = query_job.result()  # Waits for query to finish
```

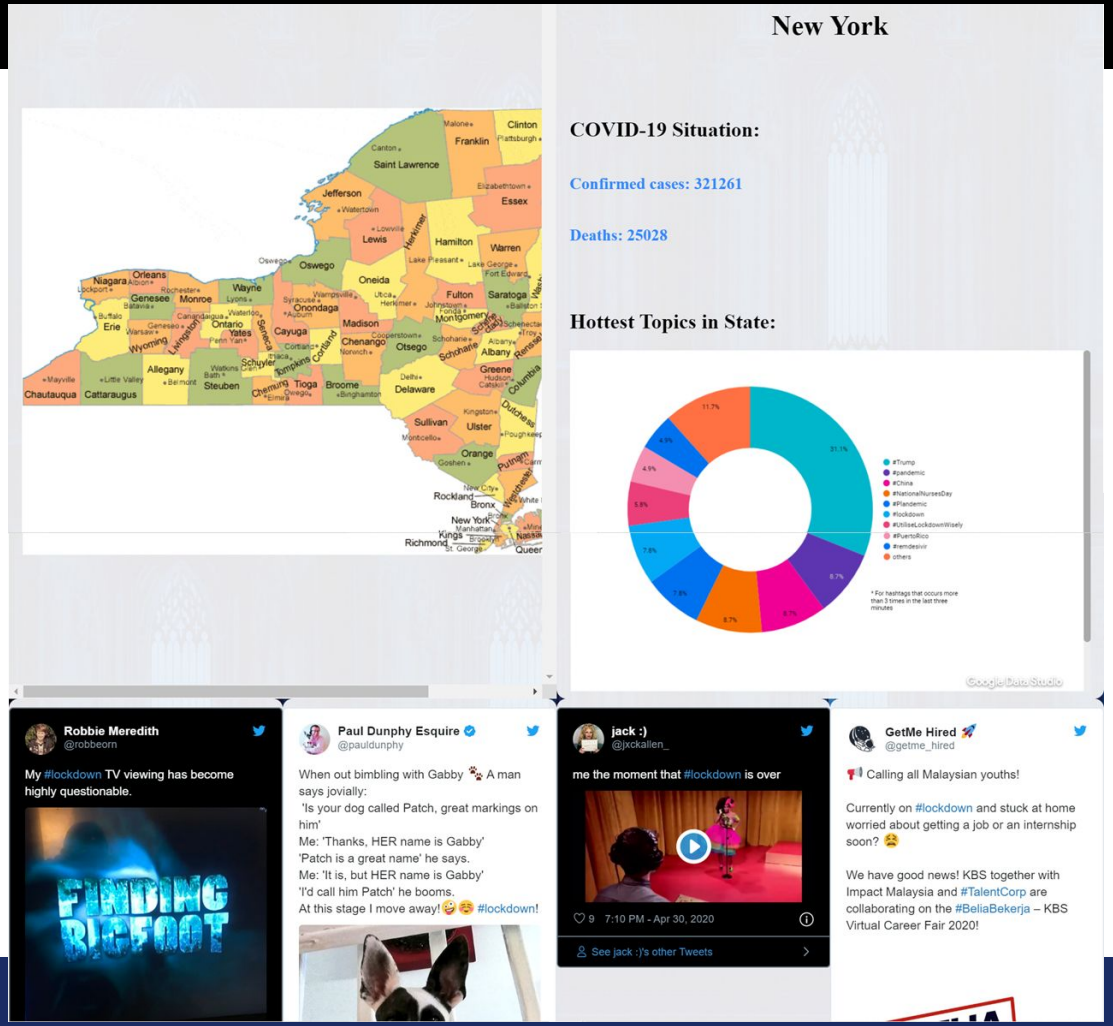
```
{'created_at': 'Thu May 07 03:31:25 +0000 2020', 'id': 1258237976255115264
{'created_at': 'Thu May 07 03:31:25 +0000 2020', 'id': 1258237975919562753
{'created_at': 'Thu May 07 03:31:24 +0000 2020', 'id': 1258237974556311553
{'created_at': 'Thu May 07 03:31:24 +0000 2020', 'id': 1258237973235187712
{'created_at': 'Thu May 07 03:31:23 +0000 2020', 'id': 1258237968847863808
{'created_at': 'Thu May 07 03:31:21 +0000 2020', 'id': 1258237960878731269
```

Use the data stored in Google Bigquery to obtain the tweets and tweets_id about the corresponding topic.

Solution

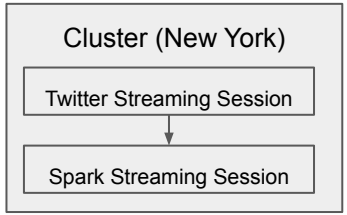


Use Django to integrate all the data and finish the visualization part.



Detail information of each state will be showed if click on the map. These information includes a more detailed map, summary of confirm cases, deaths and top 10 hottest twitter hashtags in the state, and some selected tweets in this area.

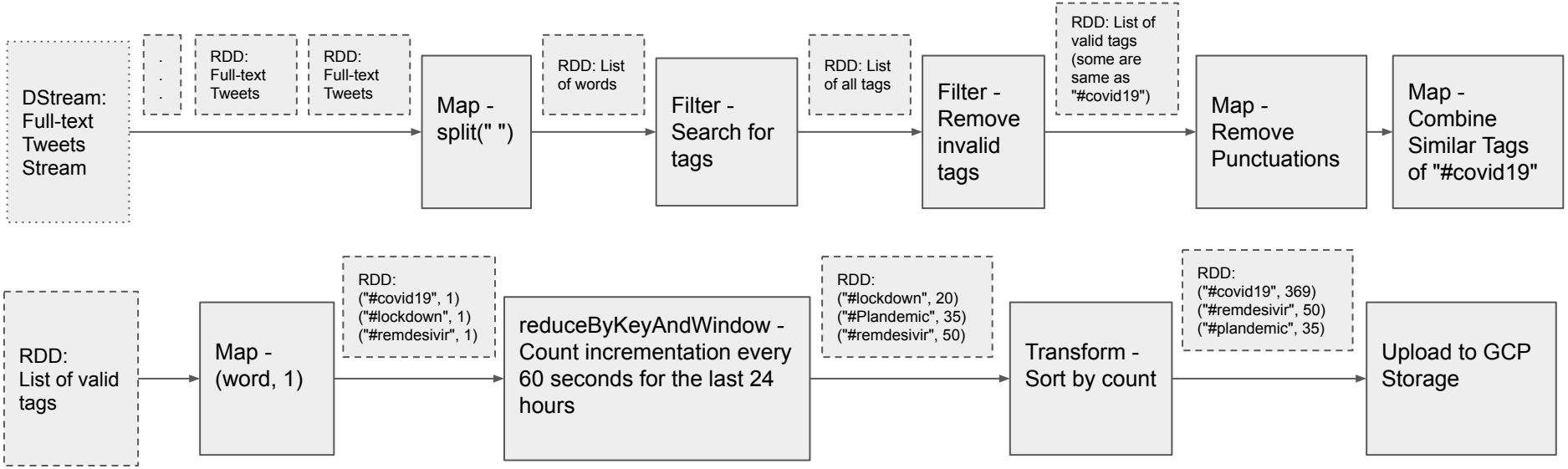
Streaming Part - Algorithm



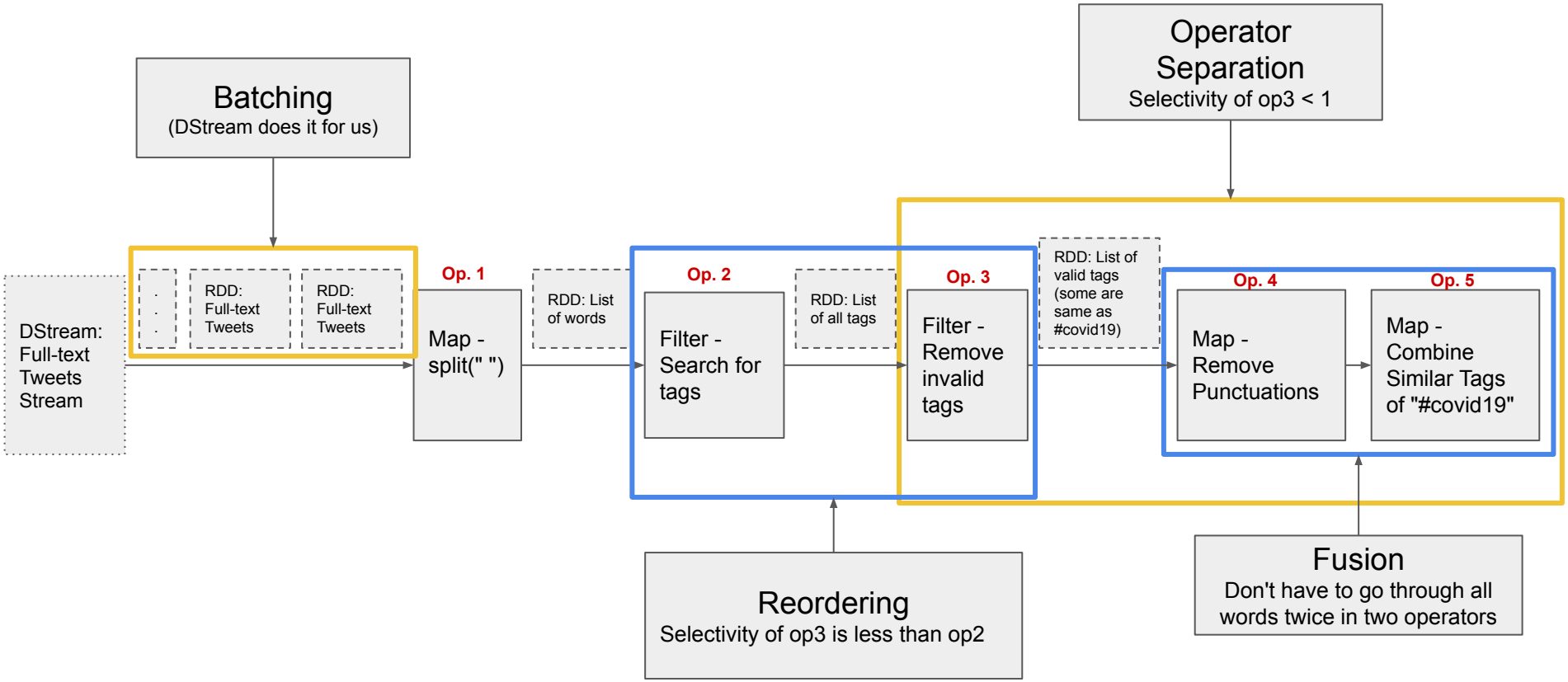
For each state, we maintain two cluster jobs on the cloud:

Job 1: getting real-time tweets, filter with states' location and tags pool, send to job 2
Job 2: analyze the tweets stream, get trending topics, send to GCP storage

Here we focus on details and optimization for job 2.



Streaming Part - Optimization



Demo / Results

Future Steps

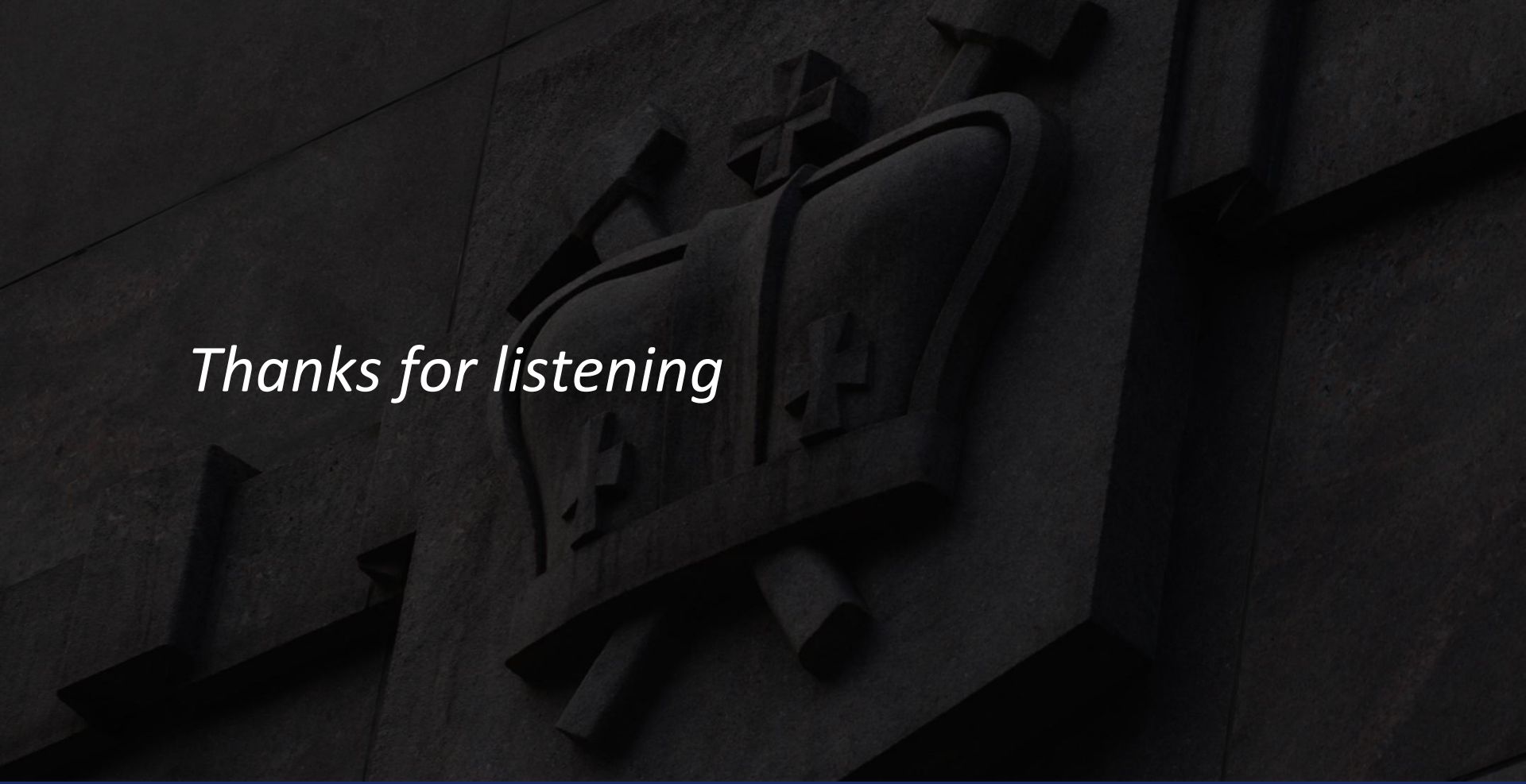
What we want to improve in the future:

1. Apply semantic analysis / topic modeling to the trending sub-topics with the corresponding tweets, so that we can learn the topics in real-time and extend the presenting of topics-related information to the news literature without #hashtags

2. Plots about US confirmed cases & deaths trend over past few months

3. More detailed information like what are people focusing on in different counties

4. Optimization on the frontend, especially try to directly “connect” it to the stream



Thanks for listening