# Government Policies
# And
# Public Sentiments

BIA-667:
Introduction to Deep Learning and Business Applications
Instructor: Rong Liu

**By- Group C**

Archana Kalburgi
Darpan Shah
Rajiv Mahajan
Siddarthan Chitra Suseendran

# Contribution of team members

| Team members | Tasks |
|---|---|
| Archana | ● Fetched 900000 tweet text from tweetids<br>● Merged datasets<br>● Implemented RNN with LSTM<br>● Implemented Bidirectional LSTM<br>● CNN, CNN + LSTM<br>● Analysis<br>● Midterm report<br>● Final report |
| Siddarthan | ● Scraped tweets belonging to 6 different categories of policies<br>● Implemented Naive Bayes, SVM models (Naive Baseline, supervised)<br>● K Means clustering (Naive model, unsupervised)<br>● Implemented CNN, CNN + LSTM, Autoencoders<br>● Midterm report,Final report |
| Rajiv | ● Midterm report<br>● Supported in data collection<br>● Final report<br>● Presentation Slides |
| Darpan | ● Analysis<br>● Midterm report<br>● Final report<br>● Presentation Slides |

# Introduction

Globally, governments have responded to the rapid growth of the Coronavirus pandemic by enacting different nationwide measures against it. In order to fight the pandemic situation, governments have framed new policies related to health testing, vaccines, mask mandates, mass gatherings and others. And public sentiment is an influential indicator of crisis response. With social media platforms, global populations have access to previously unmatched communication channels. We are using data from Twitter to gain an insight into the emotions of people over a particular category of policy.

## Problem description

Our goal here is to predict how a certain category of a policy is going to be perceived on Twitter by social media users in the future. In order to accomplish our goal, we have collected the text data (scraped from Twitter using SNScrape) to train deep learning algorithms that have been discussed throughout this coursework. In the current study, Naive Bayes was implemented as a baseline model, and the accuracy of Recurrent Neural Network(RNN), RNN with LSTM, Bidirectional LSTM, and Convolutional Neural Network(CNN) is being examined.

## Motivation

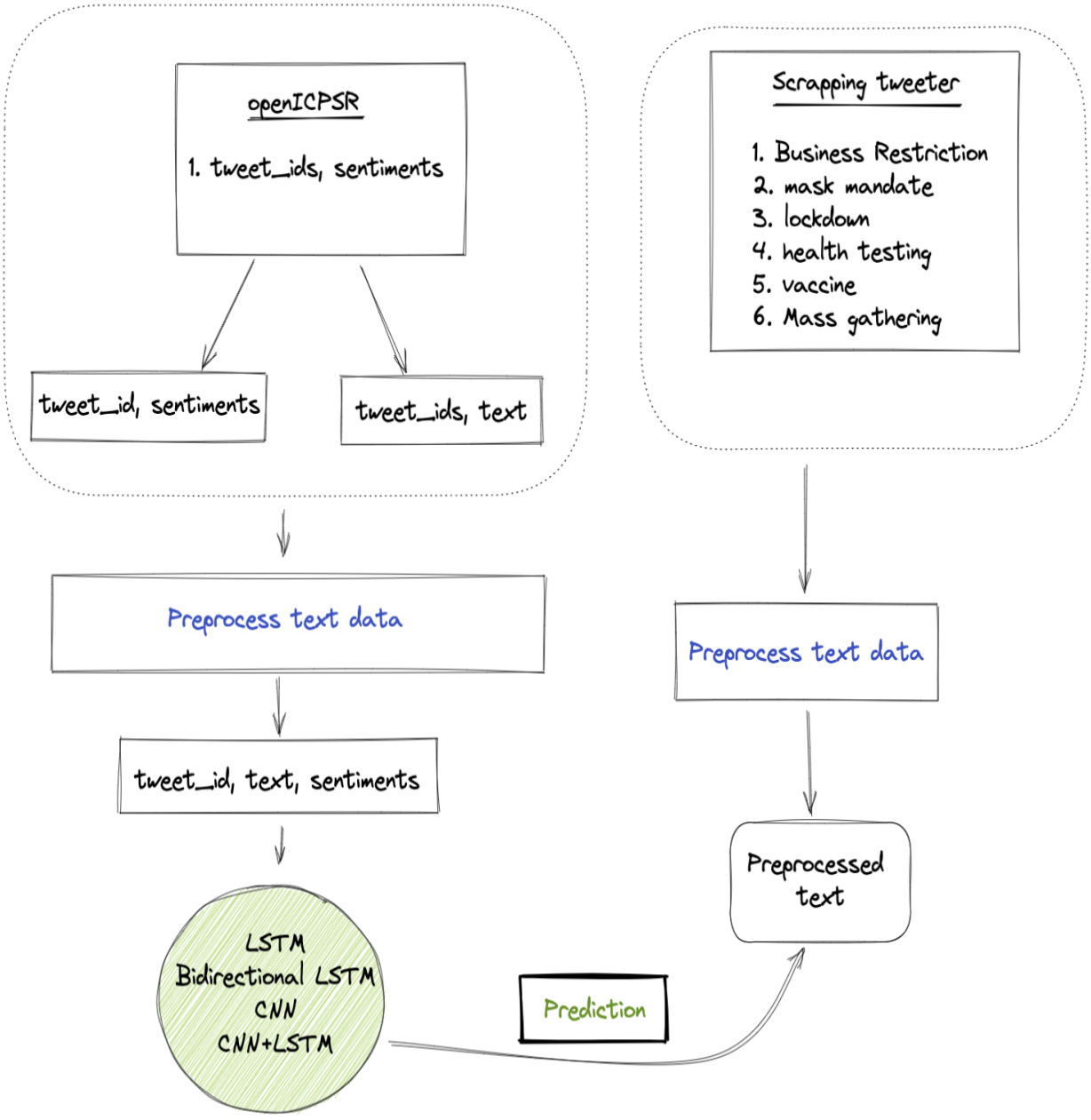
Around the world, governments have attempted to control the situation through the implementation of new policies and by amending those already in place as part of the COVID-19 containment process. It is important to understand that any changes to government policies depend on the perception that the general public holds of them. It is consistent with the truth that the virus continuously infects different countries in different time frames which then results in governments implementing a variety of response strategies and policies. Our prediction will help governments frame such policies that will give emotional security to the people of their country.

## Research questions

It has been established in the literature [3] that fear (as an emotion category) is associated with anxiety (as a mental disorder) and sadness (as an emotion category) with depression (as a mental disorder). So, it may be beneficial to study the impact of the emotion intensity scores and how they are changing over time. It is our hope that our predictions will help in examining the state of public mental health. Additionally, we

hope to find and extract features from the tweets and cluster them into groups using AutoEncoders and KNN for future research or receive any knowledge insights.

# Workflow



# Challenges and a brief survey of the state-of-the-art

## Challenges

When we set out to achieve our objective, we did not have any data that could be used to train any kind of model. We made use of two sources to collect data from.

1. First, from the Inter-university Consortium for Political and Social Research (ICPSR), openICPSR which is a self-publishing repository for social, behavioral, and health sciences research data.
2. Twitter social media platform

It became necessary for us to extract tweets (text data) using the tweet ID that we had collected from openICSPR. It was then necessary to merge this data with the data that contained the sentiment labels. By doing the above steps we were able to prepare the data that could be utilised to train our deep learning models.

Although we were able to extract the tweets using the tweet id's in the openICSPR dataset, we had to separately scrape data based on the policies since the the tweets in the openICSR dataset were generalized based on the keywords "Covid", "Corona" and "Wuhan". We achieved this by incorporating the keywords based on the policies in the snscrape module.

The challenge/disadvantage we had with this method was, the tweets had to be collected sporadically over any given time frame after the policy had been released since collecting tweets  before and after the release of a policy was possible but only a minimal amount of data could be scraped which wouldn't provide any inference at the end.

The keywords used to scrape the policy based tweets on Covid-19 in the United States were inferred from the Corona-net dataset.

Additionally as a bonus we tried to fit an unsupervised model to see if there's any patterns / feature extractions when we fit the scrapped tweets based on certain policies into an unsupervised model.

## Survey

Studies have been done using Twitter data that reveals the public opinion towards COVID- 19. For example, "Revealing Public Opinion Towards COVID-19 Vaccines With Twitter Data in the United States: Spatiotemporal Perspective"[1] reveals public opinion

towards the covid-19 vaccines using Twitter data and how the results can be used in educating vaccine skeptics.

“Evolution of public sentiments during the COVID-19 pandemic: Case comparisons of India, Singapore, South Korea, the United Kingdom and the United States” [2] suggest potential associations between government response actions both in terms of policy and communications, and public sentiment trends.

Two machine learning companies Jataware and Overton collected raw sources of information about policies. Jataware detects whether a given article is indicative of a governmental policy intervention related to COVID-19. Drawing inspiration from Jataware’s work, in this project we are attempting to answer if a policy related to COVID-19 will have one of 5 basic human emotional reactions.

# Description of dataset and data statistics

## Dataset from openICPSR

The dataset we picked was a large dataset for researchers to discover public conversation on Twitter surrounding the COVID-19 pandemic.

This was collected from 28 January 2020 to 1 September 2021. It has over 198 million Twitter posts from more than 25 million unique users using four keywords: “corona”, “wuhan”, “nCov” and “covid”. Each tweet is being labeled with five quantitative emotion attributes indicating the degree of intensity of the valence or sentiment (from 0: very negative to 1: very positive), and the degree of intensity of fear, anger, happiness and sadness emotions (from 0: not at all to 1: extremely intense), and two qualitative attributes indicating the sentiment category (very negative, negative, neutral or mixed, positive, very positive) and the dominant emotion category (fear, anger, happiness, sadness, no specific emotion) the tweet is mainly expressing.

We have taken a subset of the data described above in order to use it in the context of our study. In order to get the labeled sentiments, we have taken advantage of the data that is central to the United States which has 929337 data points. A snippet of the same can be found below:

A few rows of the merged dataset are as follows :

| | tweet_id | user_id | tweet_timestamp | keyword | \ |
|---|---|---|---|---|---|
| 0 | 1221958334661779458 | 18527874 | 2020-01-27 16-49-04 | wuhan | |
| 1 | 1221959351461720064 | 35527998 | 2020-01-27 16-53-06 | wuhan | |
| 2 | 1221959956951224320 | 415915436 | 2020-01-27 16-55-31 | wuhan | |
| 3 | 1221961233026281472 | 22586384 | 2020-01-27 17-00-35 | wuhan | |
| 4 | 1221961678058926080 | 2399087653 | 2020-01-27 17-02-21 | wuhan | |

| | valence_intensity | fear_intensity | anger_intensity | happiness_intensity | \ |
|---|---|---|---|---|---|
| 0 | 0.513 | 0.550 | 0.358 | 0.364 | |
| 1 | 0.526 | 0.387 | 0.421 | 0.364 | |
| 2 | 0.578 | 0.328 | 0.280 | 0.440 | |
| 3 | 0.497 | 0.464 | 0.431 | 0.339 | |
| 4 | 0.448 | 0.449 | 0.425 | 0.291 | |

| | sadness_intensity | sentiment | emotion |
|---|---|---|---|
| 0 | 0.344 | neutral or mixed | no specific emotion |
| 1 | 0.368 | positive | happiness |
| 2 | 0.383 | positive | happiness |
| 3 | 0.388 | neutral or mixed | no specific emotion |
| 4 | 0.474 | negative | sadness |

fig1

As illustrated in fig1, only tweet_ids were present, which are unique identifiers assigned by Twitter to each tweet. Using this information, we were able to retrieve the tweet text from Twitter[4]. There were some rows that we had to drop since they could not be located because the user had deleted those tweets. The total number of rows was 909414 after we removed the dropped rows. Following the retrieval of the tweets, we had to merge the two datasets based on the column tweet_id.

| | tweet_id | text | \ |
|---|---|---|---|
| 0 | 1222012694011629568 | The Wuhan Virus: How to Stay Safe | |
| 1 | 1222070690007920640 | @Jimmyjude13 Well I mean... | |
| 2 | 1222070757120974849 | US to expand virus screening at 20 airports fo... | |
| 3 | 1222070810581655553 | @DrOz: Reports surrounding coronavirus have be... | |
| 4 | 1222070816076107781 | Wuhan in lock down: I speak with Wayne Dupleis... | |

| | user_id | tweet_timestamp | keyword | valence_intensity | fear_intensity | \ |
|---|---|---|---|---|---|---|
| 0 | 86732334 | 2020-01-27 20-25-04 | wuhan | 0.384 | 0.572 | |
| 1 | 4379240362 | 2020-01-28 00-15-32 | wuhan | 0.460 | 0.359 | |
| 2 | 14268564 | 2020-01-28 00-15-48 | wuhan | 0.443 | 0.459 | |
| 3 | 1525512522 | 2020-01-28 00-16-00 | wuhan | 0.403 | 0.520 | |
| 4 | 27443744 | 2020-01-28 00-16-02 | wuhan | 0.503 | 0.465 | |

| | anger_intensity | happiness_intensity | sadness_intensity | sentiment | \ |
|---|---|---|---|---|---|
| 0 | 0.358 | 0.195 | 0.453 | negative | |
| 1 | 0.435 | 0.276 | 0.406 | negative | |
| 2 | 0.410 | 0.278 | 0.369 | negative | |
| 3 | 0.448 | 0.224 | 0.449 | negative | |
| 4 | 0.378 | 0.338 | 0.409 | neutral or mixed | |

| | emotion |
|---|---|
| 0 | fear |
| 1 | anger |
| 2 | fear |
| 3 | fear |
| 4 | no specific emotion |

fig2

Lengths of the tweets text over the train dataset

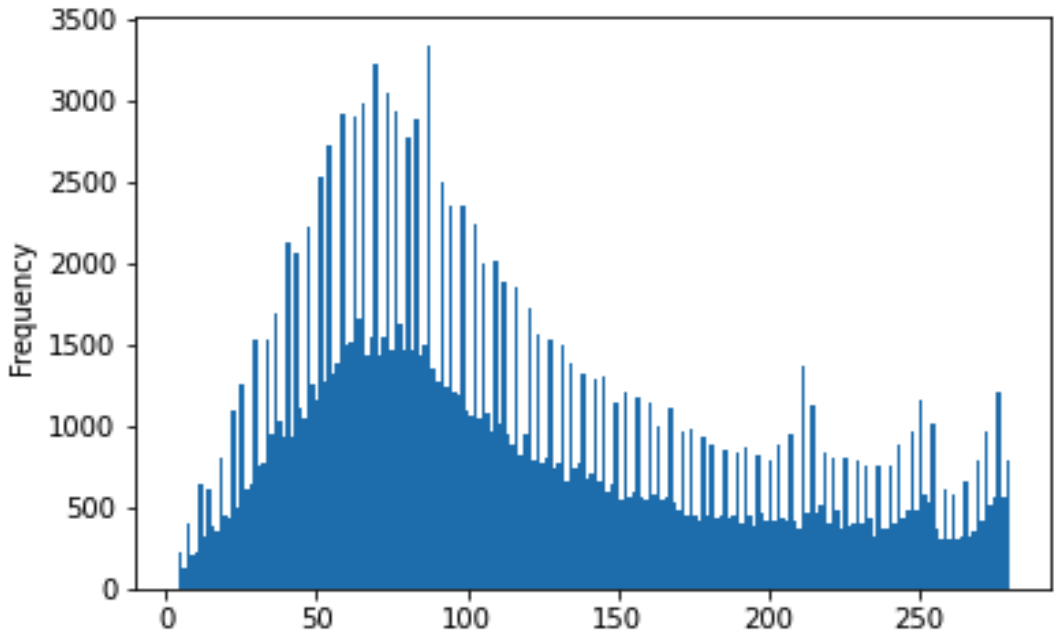



fig3

Distribution of sentiments in the train dataset:

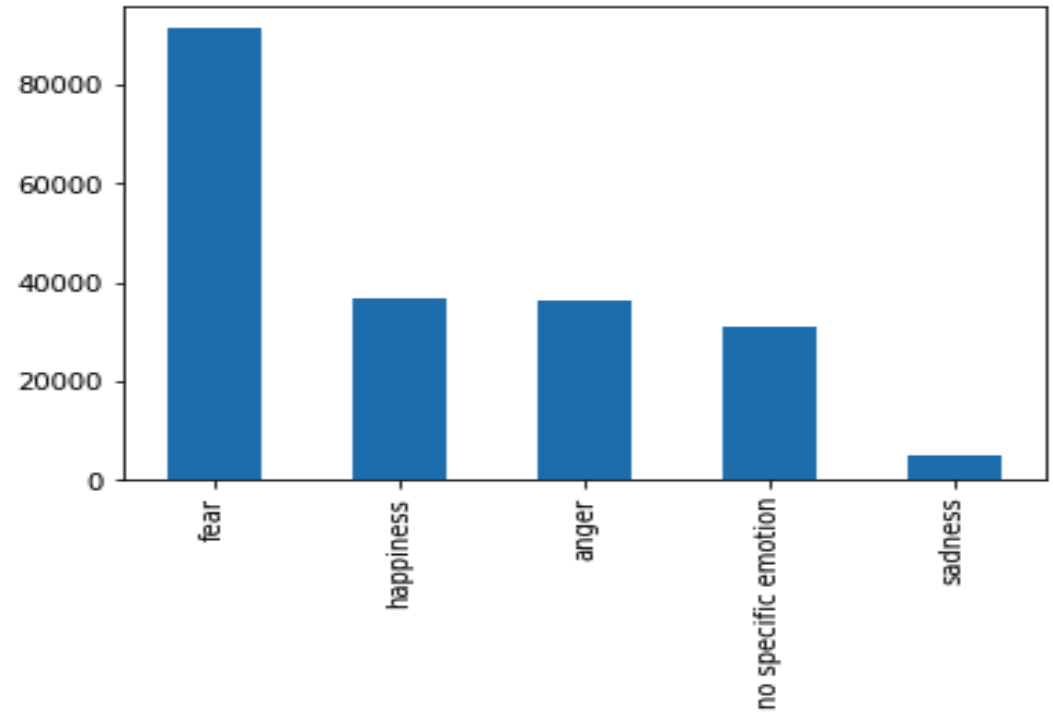



fig4

Distribution of the intensities of all the sentiments

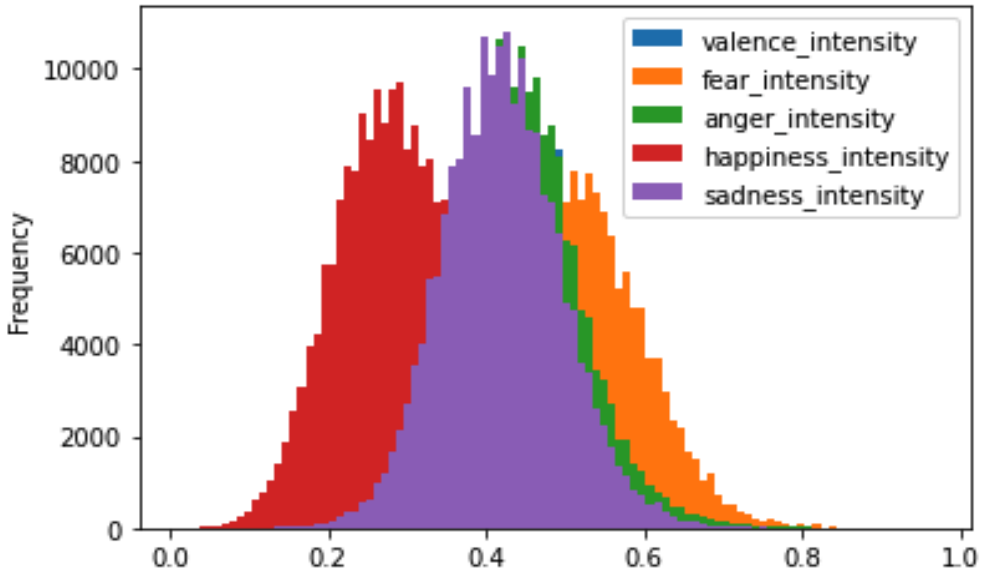



fig5

Dataset statistics:

1. openICPSR : 54,726,352
2. openicpsr-unique : 54,380,875
3. tweet_text fetched : 2,465,153
4. cleaned_tweet: : 1,001,286
5. clean_tweet_id : 929,336
6. unique_tweet: : 909,414
7. Merged Tweets and sentiments : 928,881

## Dataset from Twitter

Using SNScrape we have scrapped Twitter data and gathered the tweets related to a particular category of government policy in CSV files. We have collected the tweets belonging to the following 6 categories of policies. The number of tweets collected for each category after preprocessing is as indicated below

| | Categories | Tweets |
|---|---|---|
| 1 | Mass gathering | 5,306 |
| 2 | Mask mandate | 1,409 |
| 3 | Health testing | 5,339 |
| 4 | Business restriction | 5,410 |
| 5 | Lockdown | 5291 |
| 6 | Vaccine | 5209 |

After combining the data from all the categories we ended up with 27964 rows of data on which we will be predicting the sentiments.

| | | |
|---|---|---|
| **0** | mask_mandate | sunday afternoon mask mandat impishchimp anyon... |
| **1** | mask_mandate | cant believ vaccin mandat didnt get rid covid ... |
| **2** | mask_mandate | day 615 stop spread citizen mask mandat mask m... |
| **3** | mask_mandate | covid alway see deni mask mandat state |
| **4** | mask_mandate | addit allow children vaccin mask mandat oregon... |

fig6

Distribution of the categories in the dataset:

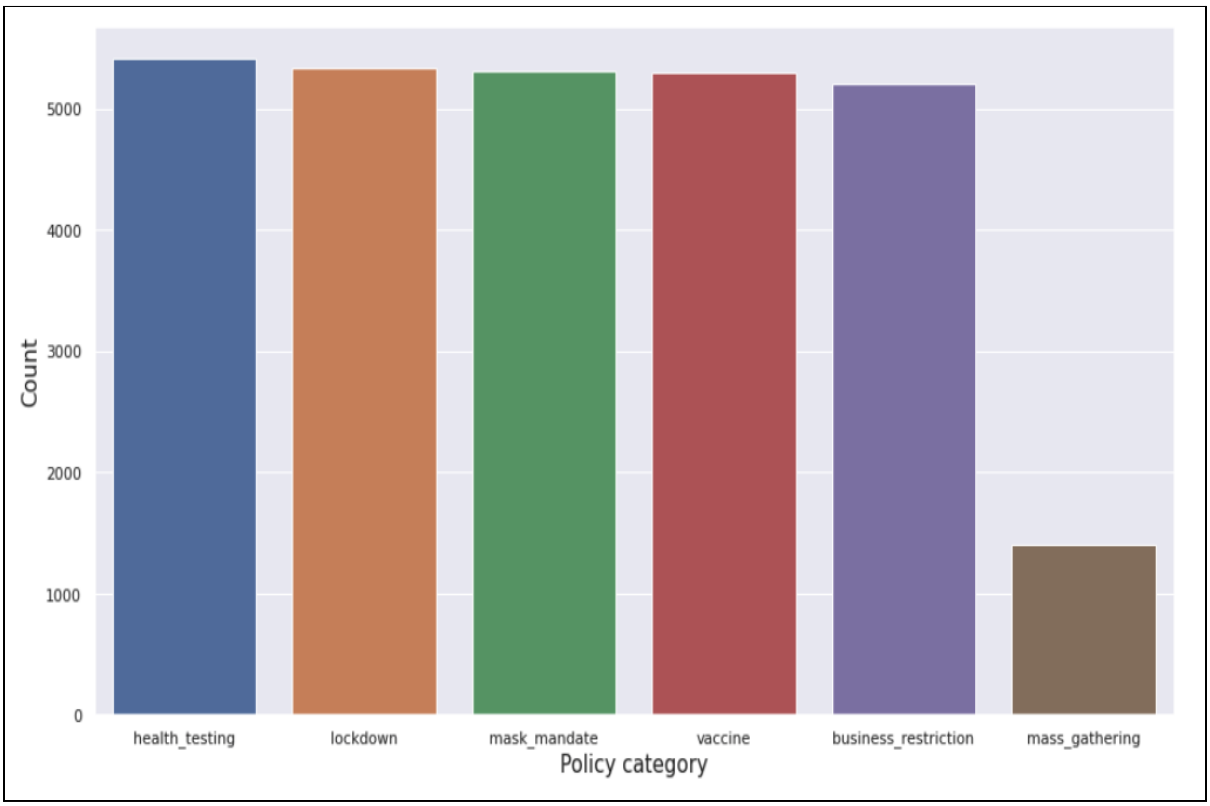



fig7

# Algorithms and baseline model for comparison

## Why are emojis eliminated?

The most popular way to embed emojis for predictions is through the use of the **emoji2vec** embedding to extract the emoticon's sentiments. In spite of the fact that emojis and emoticons are widely used in social media to convey emotions, moods, and ideas, we have eliminated them during the data preprocessing for our project.
Emojis were eliminated because studies[5] have shown that adding embeddings doesn't improve the model's performance. In fact, they observed that their model performed the worst in classifying the anger emotion.

## Models

Before predicting the sentiments, all the tweets are preprocessed by using tweet-preprocessor 0.6.0[4]. We have then tokenized the words using TF vectorizer and limited the embedding dimensions to 100.

For our project, we are using Naive Bayes and SVM as our baseline model, and we have achieved an accuracy of 63% through the method

On the combined dataset, we were able to predict the emotion of each sample using

1. Recurrent Neural Network with LSTM
3. Bidirectional LSTM
4. Convolutional Neural Network
5. Convolutional Neural Network with bidirectional LSTM

We have analysed the distribution of emotions across all the policy categories and analysed the performance of the algorithms.

## Recurrent Neural Network with LSTM

In the Long Short-Term Memory, long-term dependencies are learned (remembered) based on the input sequence.

Cell state (cell activation vector) gates comprise the LSTM unit, along with forget and output gates. These gates are responsible for controlling the update procedure of the cell state. LSTMs are defined by the H function which is defined as:

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
c_t &= f_t * c_{t-1} + i_t * \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
h_t &= o_t * \tanh(c_t)
\end{aligned}
$$

Where the *W* terms correspond to weight matrices and the *b* terms are bias vectors, *i, f, o* are the input, forget and output gates, *c* denotes cell state (activation vector), *σ* is sigmoid function and ∗ character means element-wise multiplication.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 280, 100)          10676200
_________________________________________________________________
lstm (LSTM)                  (None, 280, 256)          365568
_________________________________________________________________
dropout (Dropout)            (None, 280, 256)          0
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                82176
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 325
=================================================================
Total params: 11,128,429
Trainable params: 452,229
Non-trainable params: 10,676,200
_________________________________________________________________
```

## Bidirectional LSTM

It is a common practice to use Bidirectional LSTM (BiLSTM) for sequential data such as ours. The BiLSTM consists of two LSTMs. One LSTM processes the input sequence from the first element and produces an output vector. The second LSTM processes the input sequence in reverse order and also produces output vectors. Both output vectors have dimension D. The final output vector from BiLSTM with dimension 2D is then created by concatenating two output vectors.

Our major inspiration for applying bidirectional LSTM was a work done in Emotion Prediction in Tweets with Bidirectional Long Short-Term Memory Neural Network[5]. The team implemented a BiLSTM and were able to achieve an accuracy of 0.657.

We trained our model using mini-batches of size 1000 for 50 epochs and we used the Adam optimizer. As an activation function in the BiLSTM and in the dense layers, we used a Rectified Linear Unit (ReLU). Dropout of 0.3 is used for the recurrent connections in the BiLSTM layer and in all dense layers. We trained the model on the openICSPR and we evaluated the trained model on the tweets of all the policy categories that we had initially collected and preprocessed. We experimented with different settings of the hyperparameters (mini-batch, dropout size etc.) but the mentioned settings showed to be the best one on the development data. These hyperparameter settings were also used for final submission

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 280, 100)          10676200
_________________________________________________________________
bidirectional (Bidirectional (None, 280, 128)          84480
_________________________________________________________________
dropout_1 (Dropout)          (None, 280, 128)          0
_________________________________________________________________
bidirectional_1 (Bidirection (None, 128)               98816
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 325
=================================================================
Total params: 10,868,077
Trainable params: 191,877
Non-trainable params: 10,676,200
_________________________________________________________________
```

## Convolutional Neural Network

We trained our model using mini-batches of size 1000 for 50 epochs and we used the Adam optimizer. We used a Rectified Linear Unit (ReLU) as an activation function in the convolutional layer. We have used 32 filters with a kernel size of 3. A pooling function is applied to each feature map to get a fixed length vector, and max pooling is then applied to extract features. The final classification is determined by a softmax function.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 280, 100)          10676200
_________________________________________________________________
conv1d (Conv1D)              (None, 280, 32)           9632
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 140, 32)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 140, 64)           6208
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 70, 64)            0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 70, 128)           24704
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 35, 128)           0
_________________________________________________________________
flatten (Flatten)            (None, 4480)              0
_________________________________________________________________
dense (Dense)                (None, 128)               573568
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 645
=================================================================
Total params: 11,290,957
Trainable params: 614,757
Non-trainable params: 10,676,200
_________________________________________________________________
```

## Convolutional Neural Network with Bidirectional LSTM

We trained our model using mini-batches of size 1000 for 50 epochs and we used the Adam optimizer. We used a Rectified Linear Unit (ReLU) as an activation function in the convolutional layer. We have used 32 filters with a kernel size of 3. A dropout of 0.3 has been set to the 2 bidirectional layers. Added dropout helped overfighting of the model compared to a model without any requlariser.

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 280, 100)          10676200
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 280, 32)           9632
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 140, 32)           0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 140, 64)           6208
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 70, 64)            0
_________________________________________________________________
bidirectional_2 (Bidirection (None, 70, 128)           66048
_________________________________________________________________
dropout_3 (Dropout)          (None, 70, 128)           0
_________________________________________________________________
bidirectional_3 (Bidirection (None, 128)               98816
_________________________________________________________________
dropout_4 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_7 (Dense)              (None, 5)                 325
=================================================================
Total params: 10,865,485
Trainable params: 189,285
Non-trainable params: 10,676,200
_________________________________________________________________
```

## Auto Encoders with K-Means

An autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data. The encoding is validated and refined by attempting to regenerate the input from the encoding. The autoencoder learns a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore insignificant data (noise).

k-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

```
Model: "model_12"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_9 (InputLayer)        [(None, 280)]             0

 dense_55 (Dense)            (None, 200)               56200

 dense_56 (Dense)            (None, 500)               100500

 dense_57 (Dense)            (None, 500)               250500

 dense_58 (Dense)            (None, 2000)              1002000

 dense_59 (Dense)            (None, 10)                20010

 dense_60 (Dense)            (None, 2000)              22000

 dense_61 (Dense)            (None, 500)               1000500

 dense_62 (Dense)            (None, 280)               140280

=================================================================
Total params: 2,591,990
Trainable params: 2,591,990
Non-trainable params: 0
_________________________________________________________________
```

# Results and analysis

The following plots illustrate the results of the above models, which were trained with 500000 samples of training samples.

## Performance of LSTM

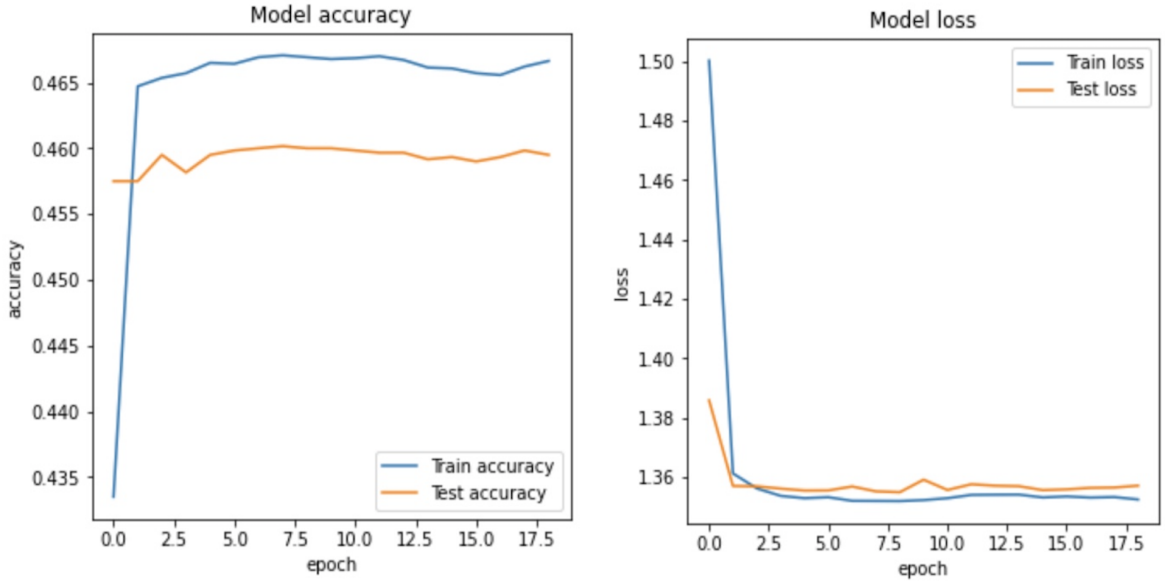



fig8

## Performance of Bidirectional LSTM

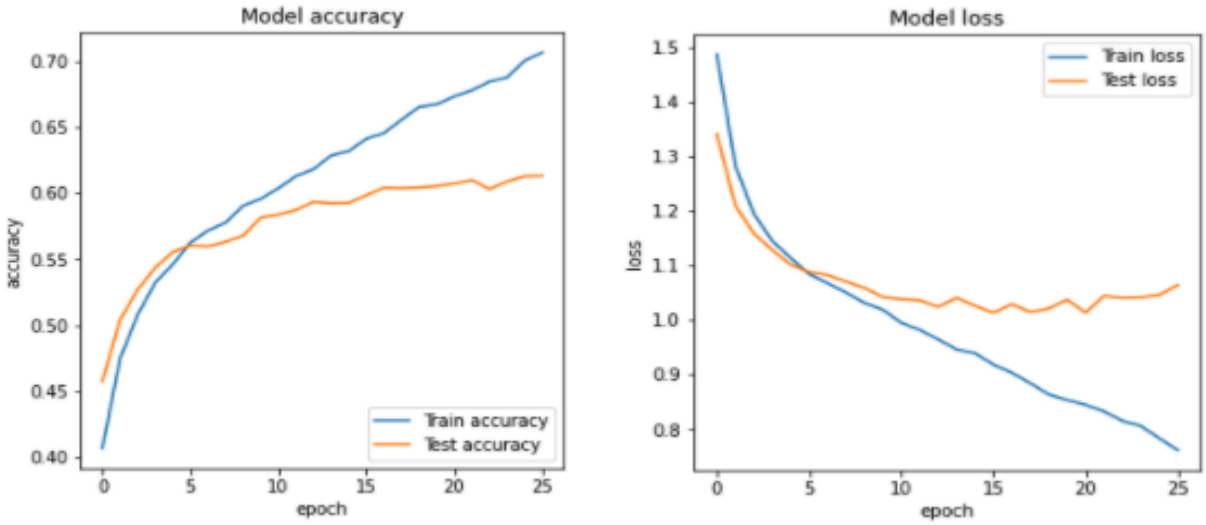



fig9

## Performance of Convolutional Neural Network

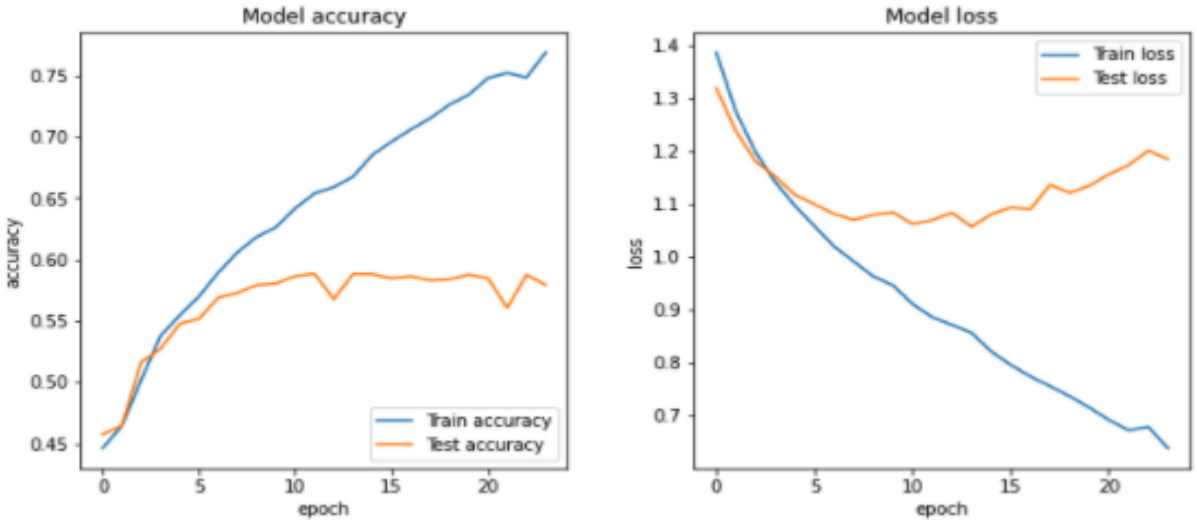



fig10

## Performance of CNN with bidirectional LSTM

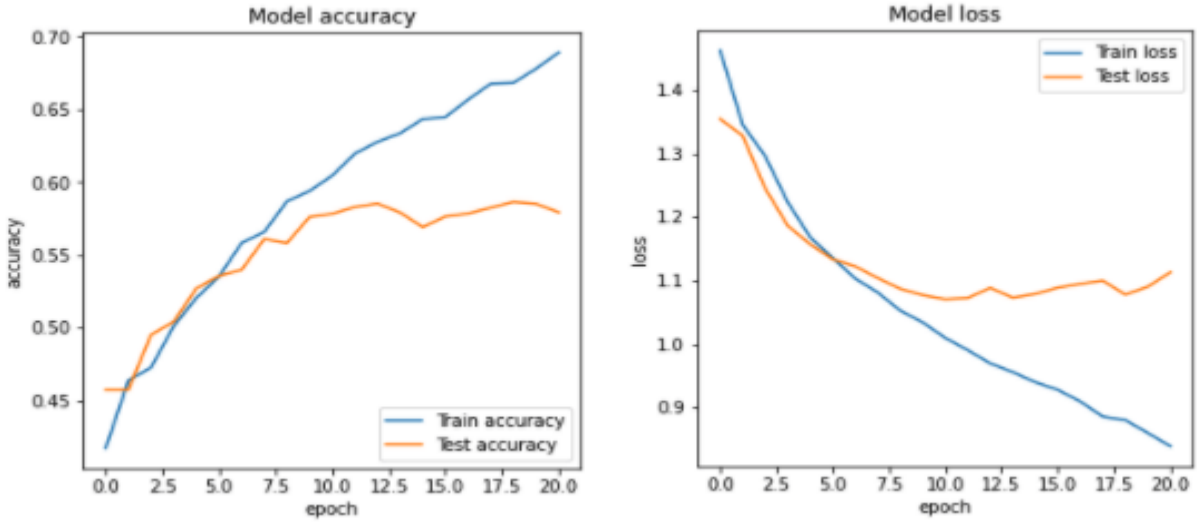
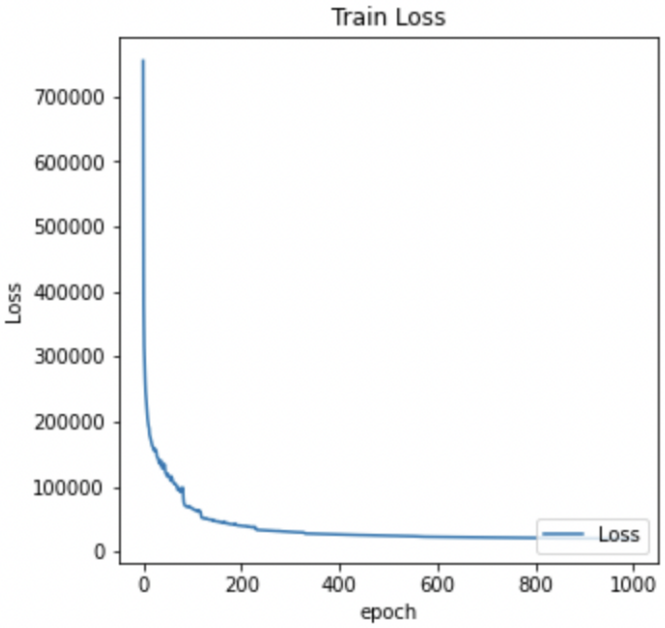
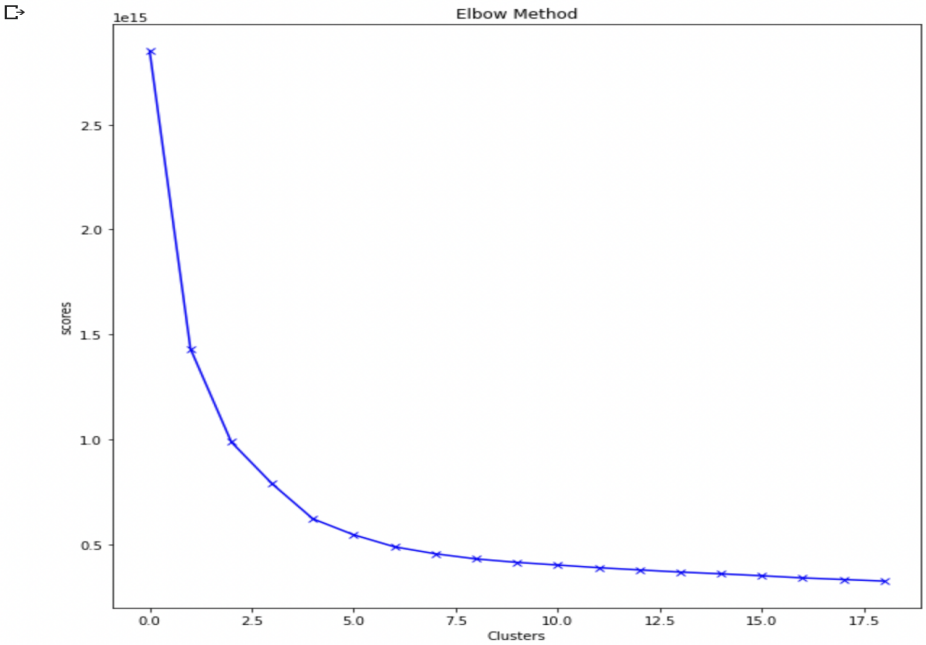
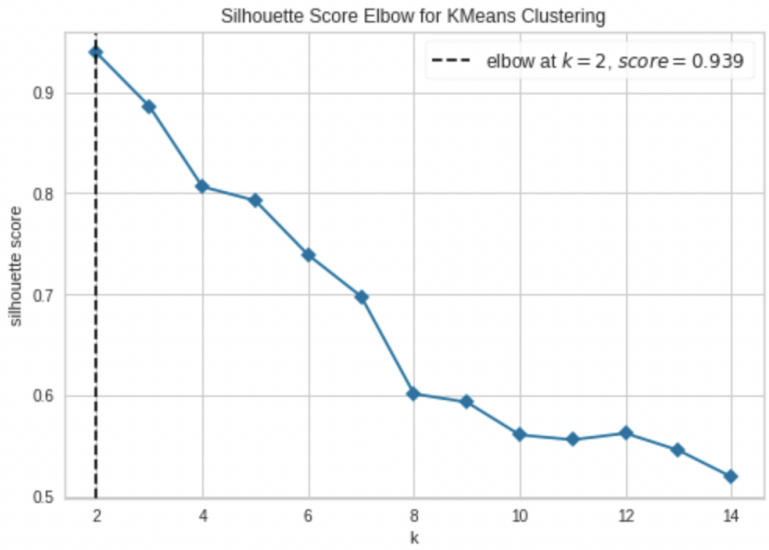



fig11

# Performance of Unsupervised Model



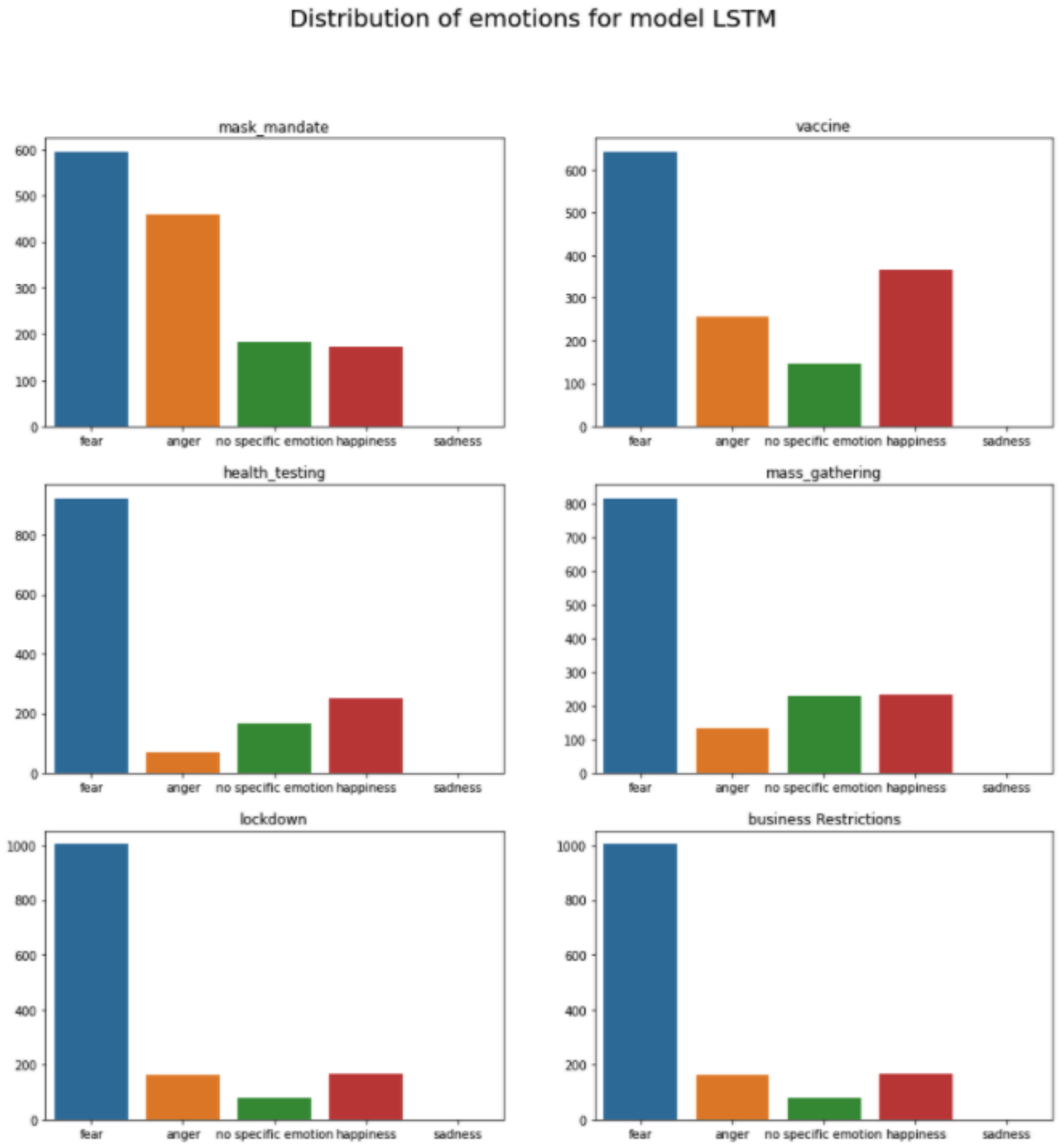





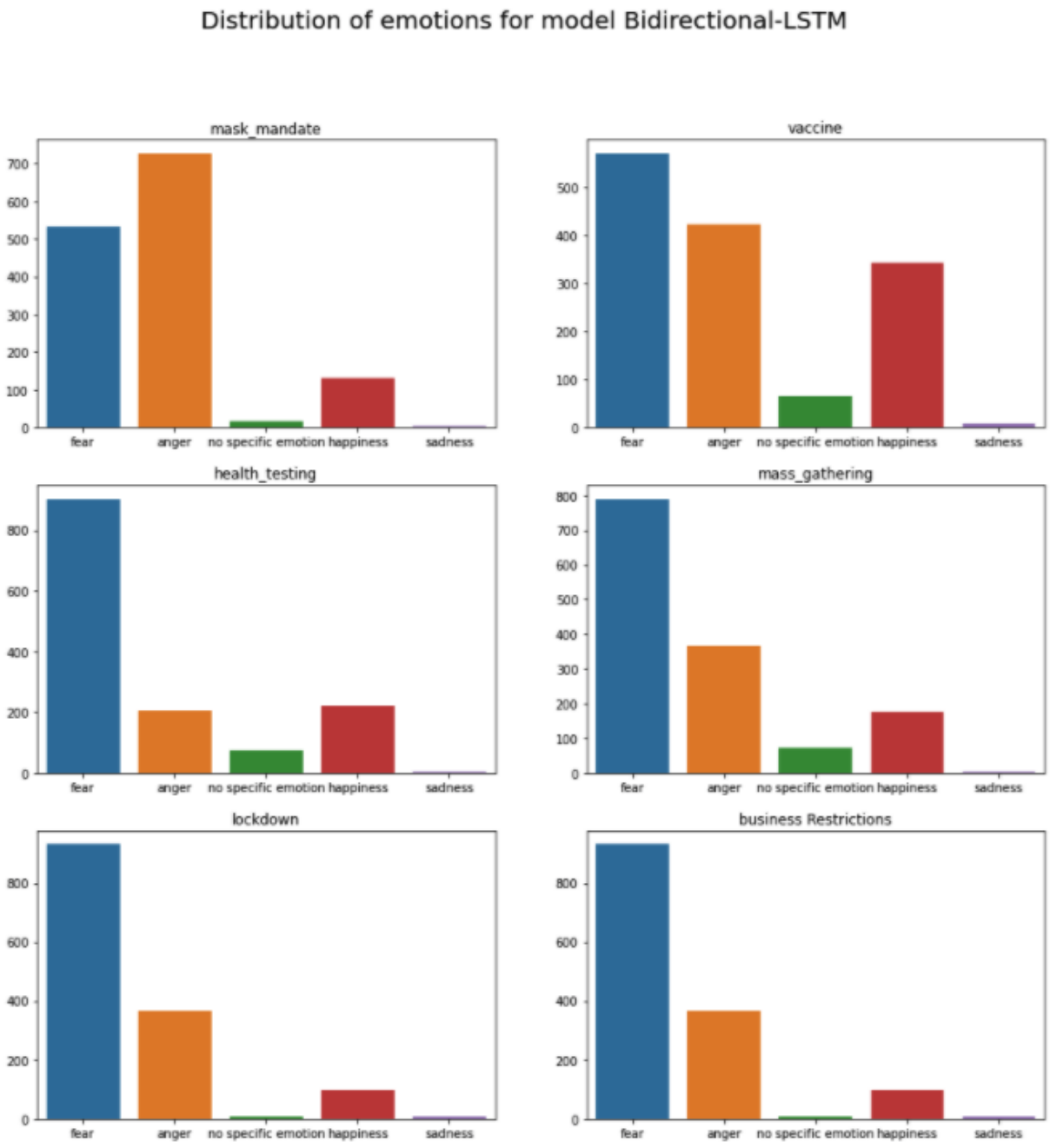

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1638599710>
```

# Model predictions of emotions

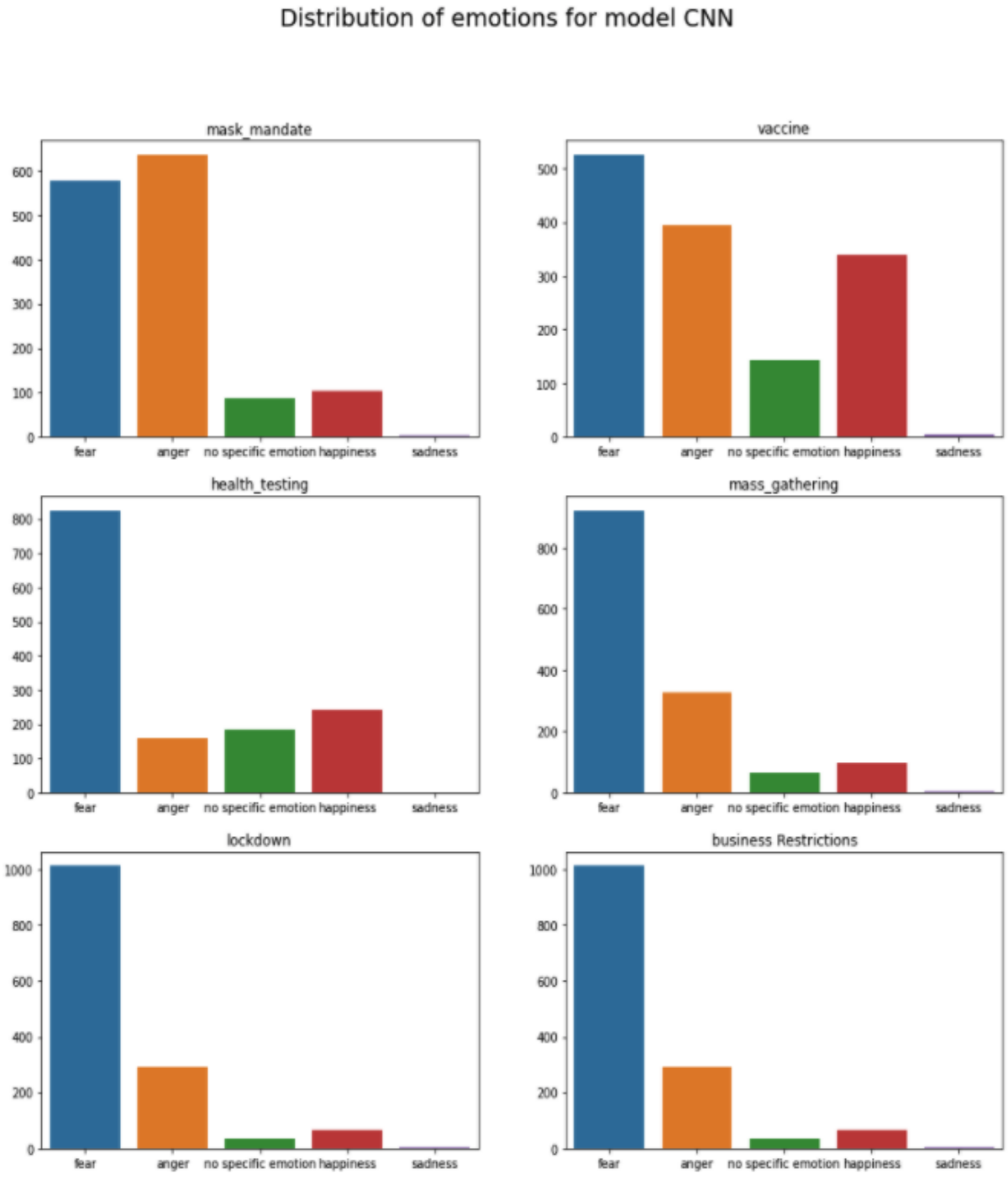
1. LSTM



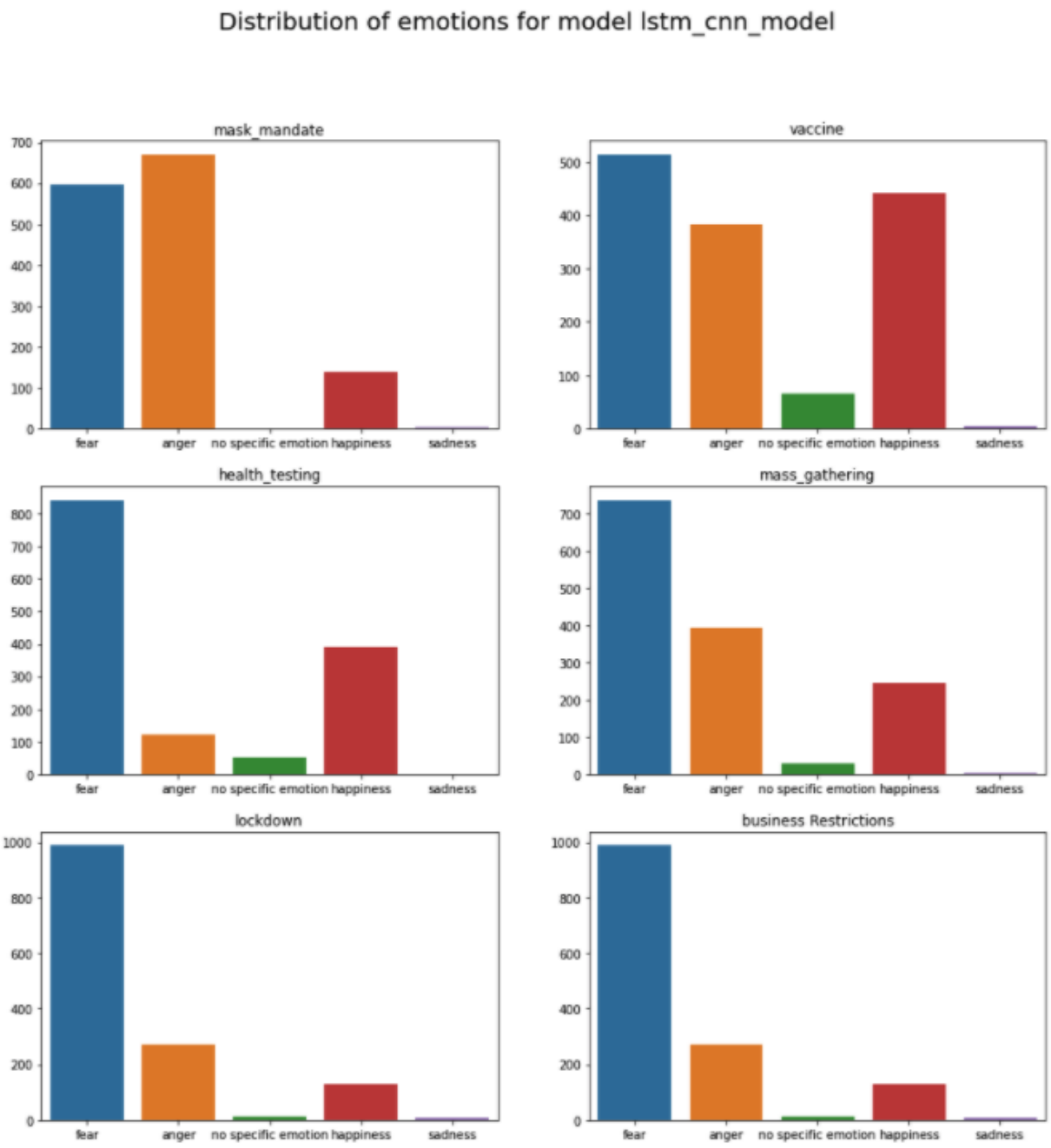
2. BiLSTM


Distribution of emotions for model Bidirectional-LSTM
mask_mandate
vaccine
health_testing
mass_gathering
lockdown
business Restrictions
fear
anger
no specific emotion
happiness
sadness

3. CNN

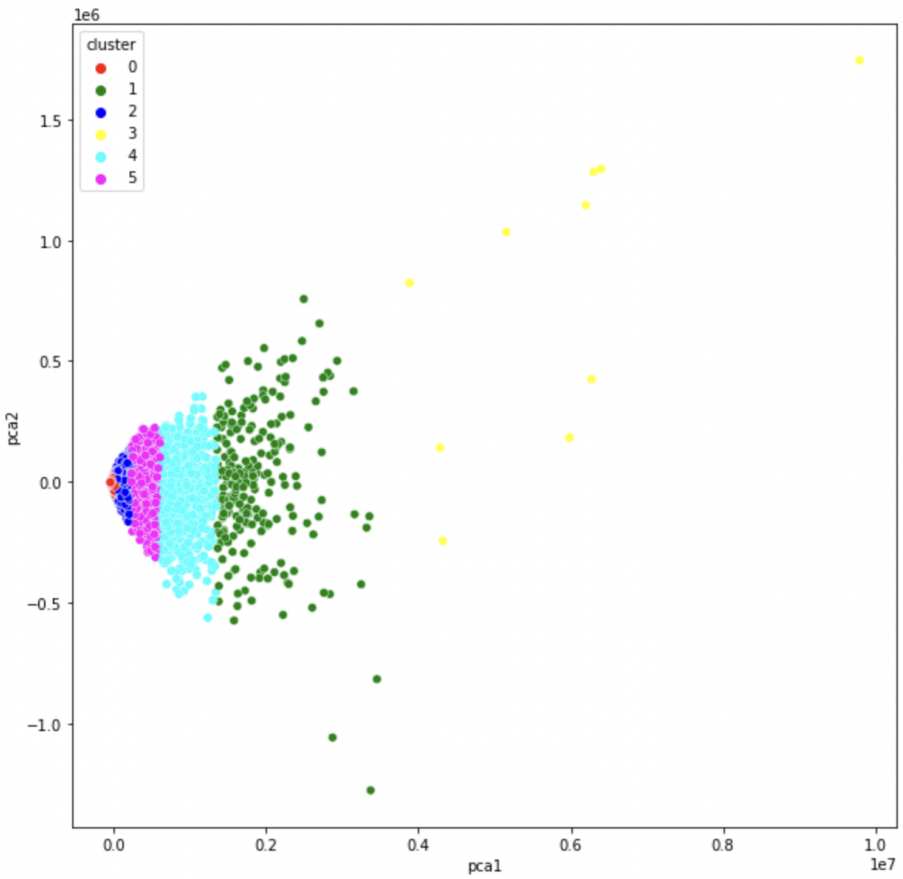

Distribution of emotions for model CNN
mask_mandate
vaccine
health_testing
mass_gathering
lockdown
business Restrictions
fear
anger
no specific emotion
happiness
sadness

4. CNN with Bidirectional LSTM




5. Autoencoder

We were significantly able to improve the silhouette score after combining K-means with auto-encoders (The optimal amount of clusters is 6). Additionally for visualization sake, we'd reduced the dimension to two using PCA to visualize the clusters.




| | emotion | text | cluster |
|---|---|---|---|
| **0** | 0.0 | wuhan virus stay safe | 0.0 |
| **1** | 1.0 | well mean | 0.0 |
| **2** | 0.0 | us expand virus screen 20 airport visitor chin... | 0.0 |
| **3** | 0.0 | report surround coronavirus chang rapid need k... | 0.0 |
| **4** | 2.0 | wuhan lock speak wayn dupleiss canadian citize... | 0.0 |

# Our analysis

## Analysis I

As a result of our analysis, we were able to see a few interesting patterns in the sentiment of the public over the 6 general categories of policies. Across all 6 categories of policies, the most dominant emotion expressed by people is **'fear'**, followed by **'anger'**. With regards to the mask mandate, we see that anger is the dominant sentiment among the people of the United States of America followed by happiness. We see very similar patterns of emotions among the people in regards to lockdown and business restrictions. More people have expressed anger towards policies related to mass gatherings compared to policies on health testing. Interesting observation is that people are more likely to be happy about vaccine policies than any other type of policy.

The fact that out of all the 30000 tweets we predicted sentiments on, only a few contained no specific emotion adds validity to our analysis

## Analysis II

| Model | Train accuracy | Validation accuracy |
|---|---|---|
| LSTM | 0.4649 | 0.4575 |
| **BiLSTM** | **0.6474** | **0.6126** |
| CNN | 0.6373 | 0.5830 |
| CNN+BiLSTM | 0.6238 | 0.5913 |

# Conclusion and future work

Throughout this research project, we have analysed the sentiments of tweets from all over the United States. A similar analysis can also be performed on tweets from other countries as well.

While we have assessed the effects of a particular category of policy, we can continue our analysis of the impact of a single policy rather than the consequences of a category of policies.

In the future, we might want to attempt another methodology utilizing a Twitter explicit language model to foresee probabilities for every emotion class for the missing objective feeling word in the given information. These probabilities could be utilized as information highlights in our model.

Our work proposes a method by which we can construct a convolutional neural network for taking a look at sentiments based on text. Research can be conducted further, such as considering the use of the word2vec tool and multilayer convolutional neural networks

The analysis of a larger number of training data sets and other types of situation or status analysis can be done.

# References

[1] Revealing Public Opinion Towards COVID-19 Vaccines With Twitter Data in the United States: Spatiotemporal Perspective

[2] Evolution of public sentiments during the COVID-19 pandemic: Case comparisons of India, Singapore, South Korea, the United Kingdom and the United States

[3] Öhman, A., Fear and anxiety. Handbook of Emotions, 709-729. (2008)

[4] Twitter preprocessor

[5] Emotion Prediction in Tweets with Bidirectional Long Short-Term Memory Neural Network