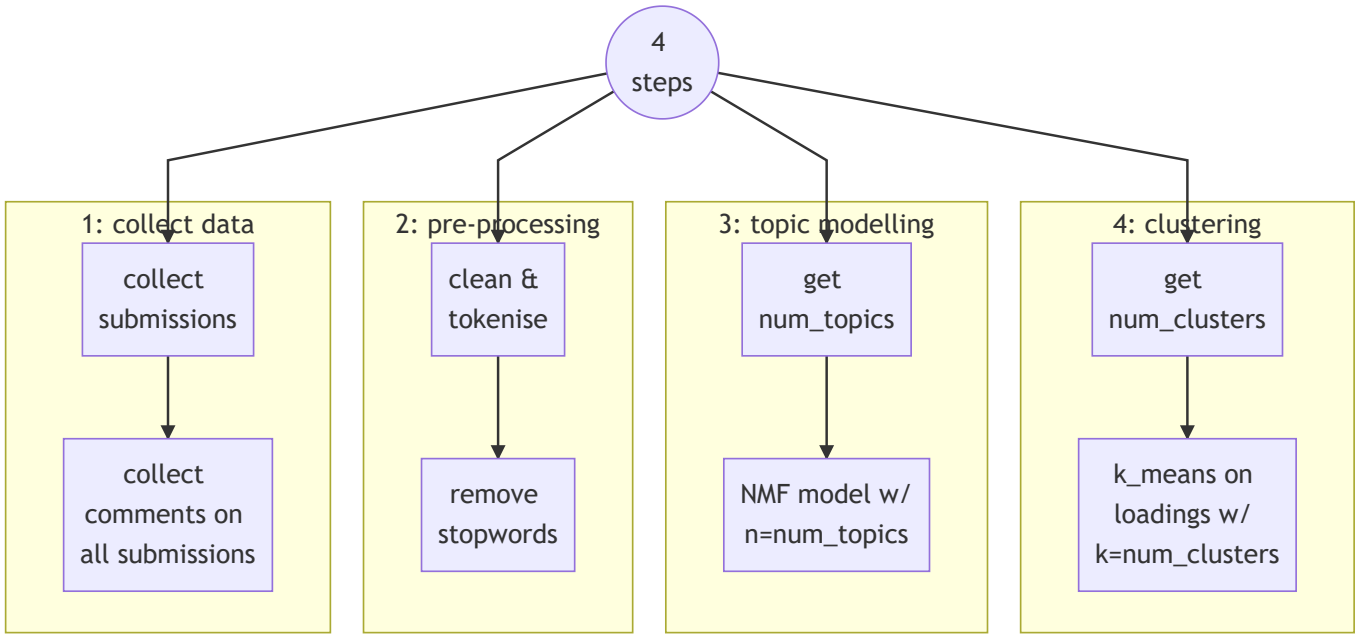
# The right opinion? Anti-vaccine conspiracies on Reddit : Technical write-up


- by Maheep Tripathi, Sanchi Singh
- for POL20200 Computational Social Science (WiSe2020) taught by Prof. Juergen Pfeffer

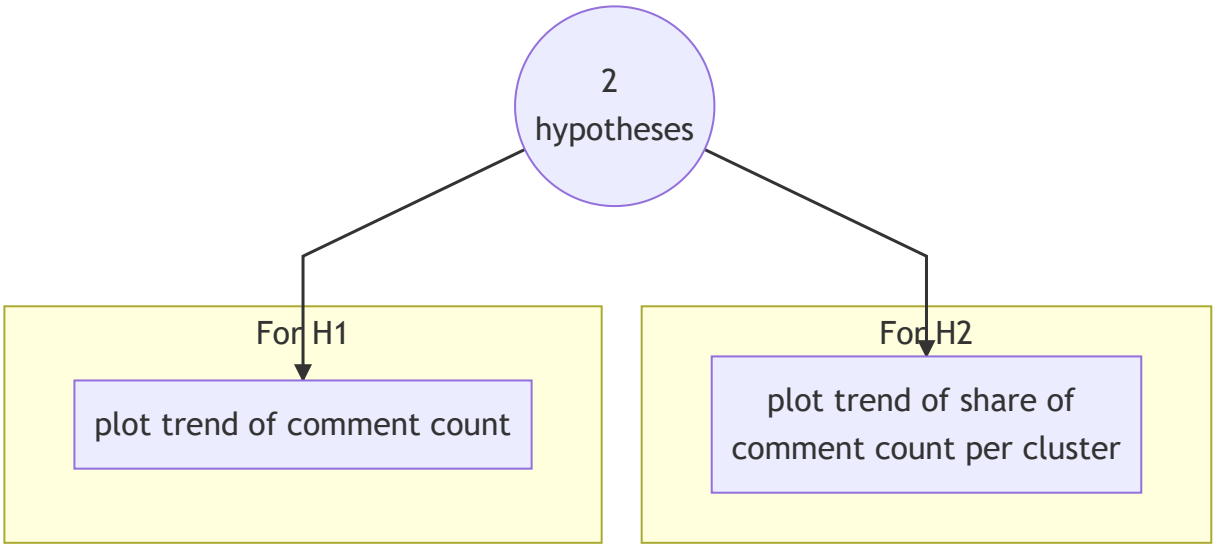

We used Python 3 for our topic analysis[1] of the subreddit r/conspiracy. We tested the following 2 hypotheses.

> H1: The COVID-19 pandemic has led to an increase in consumption of anti-vaccine content with time.
> H2: There are shifts in anti-vaccine rhetoric or concerns with time.

The flowchart below summarises our methodolgy:




We plot comment frequency twice to test our hypotheses:




Now, we look at each of the 4 steps in detail along with code snippets.

## 1. Collect Data

We used the Pushshift dataset and PRAW for data collection.

### 1.1 Collect submissions

Used a pushshift API query to get all r/conspiracy submissions containing the word 'vaccine' in 2020 (from 01 Jan 2020 to 15 Dec 2020).
Can't use PRAW (The Python Reddit API Wrapper) to get historical data, as it is limited to latest ~1000 submissions.

```
def getPushshiftData(query, after, before, subreddit):
    url = 'https://api.pushshift.io/reddit/search/comment/?
q='+str(query)+'&size=1000&metadata=true&after='+str(after)+'&before='+str
(before)+'&subreddit='+str(subreddit)
    #Print URL to show user
    print(url)
    #Request URL
    r = requests.get(url)
    #Load JSON data from webpage into data variable
    data = json.loads(r.text)
    #return the data element which contains all the submissions data
    return data['data'], data['metadata']

query = "vaccine"
subreddit = "conspiracy"
after = "1577836800" #Submissions after this timestamp (1577836800 = 01
Jan 20)
before = "1607990400" #12/15/2020 @ 12:00am (UTC)
data, metadata = getPushshiftData(query, after, before, subreddit)
```

### 1.2 Collect comments

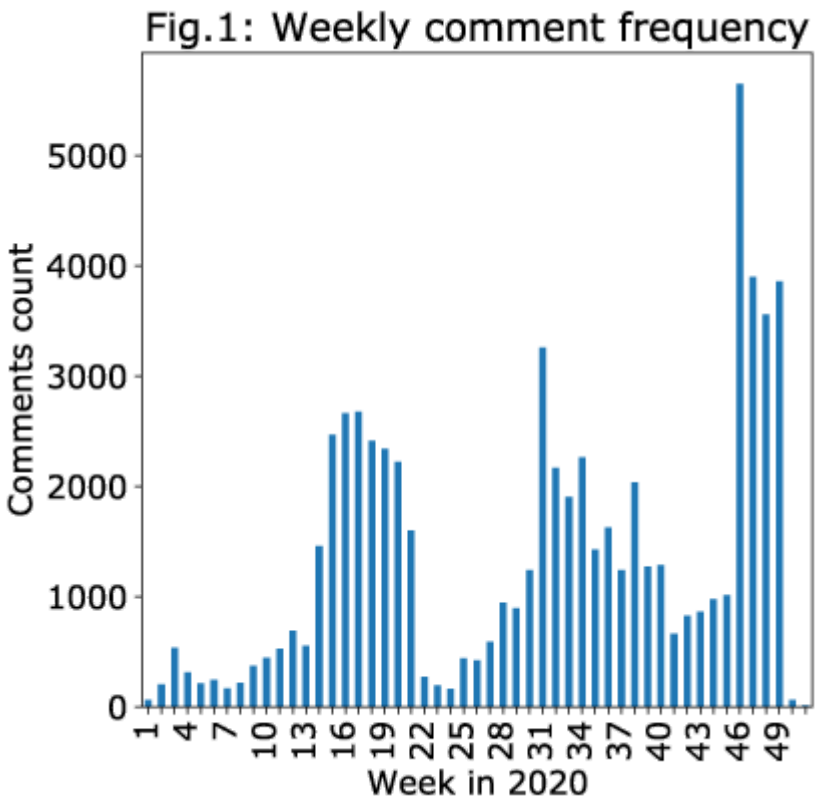
Used PRAW to iterate over all submissions and get all comments under each submission. Could've used the pushshift dataset for comments, but its coverage is reportedly lower[2].

```
import praw
reddit = praw.Reddit(client_id = my_client_id, client_secret =
my_client_secret, user_agent = my_user_agent)
```

```
# get vaccine submission IDs from pusshift csv, then iterate
for sub_id in list_subID:
    comments_data = list() #list to store data points
    submission = reddit.submission(id=sub_id)
    submission.comments.replace_more(limit=None)
    for c in submission.comments.list():
        #print(comment.body)
        name = c.author.name if c.author else '[deleted]' # handle deleted
comments
        comments_data.append((c.submission.id, c.id, c.body,
datetime.datetime.fromtimestamp(c.created_utc), name, c.score,
c.permalink))
```

## For H1

Plot weekly comment frequency to test H1. Fig.1 does not show a monotonic increase in comment frequency.



## 2. Pre-processing

### 2.1 Clean-up

1. Remove comments -
    1. where user and text are '[deleted]'
    2. where user is a bot[3]
2. Clean - removes URLs, Hashtags, Mentions, Reserved words (RT, FAV), Emojis, Smileys (Source)

```
from preprocessor import clean
```

3. Expanding contractions
4. Tokenise (make lowercase, remove punctuation)

```
from nltk.tokenize import TweetTokenizer
```

5. Miscellaneous clean-up
    1. Fix repetition in words e.g. "Cooooolll!" => "Cool!"
    2. Remove tokens that are integers
6. Remove stopwords

```
from nltk.corpus import stopwords, wordnet
stop_words = stopwords.words('english')
```

7. Stemming (not lemmatising as it needs the whole sentence context to be effective)

```
from nltk.stem import PorterStemmer
```

## 3. Topic modelling

### 3.1 Tf-idf representation

First, compute frequently occuring bigrams and trigrams, concatenate and add back to original tokenised data.

```
# Compute bigrams and trigrams
from gensim.models import Phrases
```

Represent the document-feature space with a bag-of-words model with tf-idf vectors. Tf-idf normalises term frequency such that it better represents distinctiveness - a term gets a high score if it occurs infrequently across the corpus but frequently within the document.

```
from sklearn.feature_extraction.text import CountVectorizer,
TfidfTransformer
cv = CountVectorizer(min_df=20, max_df=1.0, ngram_range=(1,1),
                     token_pattern=None, tokenizer=lambda doc: doc,
                     preprocessor=lambda doc: doc)
cv_features = cv.fit_transform(df_com_pp['text']) #Learn the vocabulary
dictionary (fit) and return document-term matrix (transform)
transformer = TfidfTransformer(smooth_idf=False);
cv_features = transformer.fit_transform(cv_features);
```

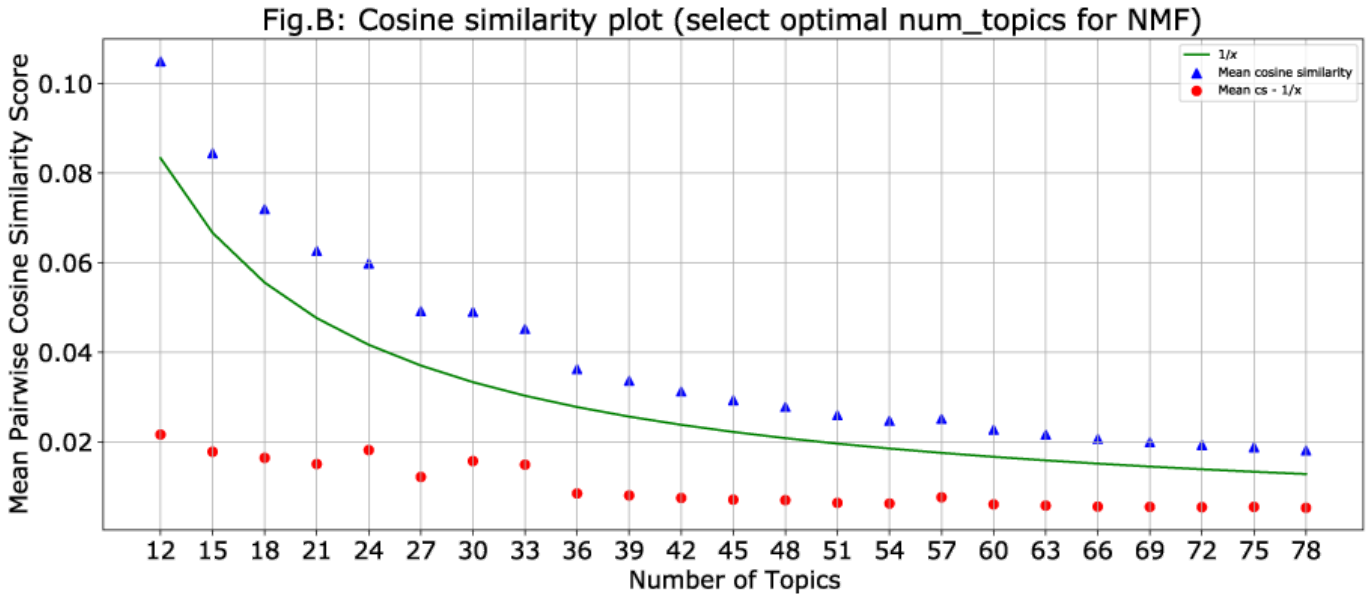
### 3.2 Optimise num_topics for Non-negative Matrix Factorisation (NMF)

```
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.decomposition import NMF
# Fucntion to find the best number of topics using cosine similarity
score.
def test_NMFs(lower_topics, upper_topics, step_size):
    num_models = (upper_topics - lower_topics) / step_size
    # Iterate through topic numbers to find an optimal number of topics
    mean_sim_scores = []
    models = {}
    count = 1
    for i in range(lower_topics, upper_topics, step_size):
        print('Testing model', count, 'of', num_models)
        count += 1
        nmf_model = NMF(n_components=i, init='nndsvd', max_iter=2000)
        nmf_model.fit_transform(cv_features)
        topic_terms_matrix = nmf_model.components_
        sim_matrix = cosine_similarity(topic_terms_matrix)# pairwise
similarity scores between topic-term vectors
        mean_sim_scores.append(np.mean(sim_matrix))# mean of similarity
scores
        models[str(i)] = nmf_model

    return models, mean_sim_scores

model_objs_NMF, sim_scores = test_NMFs(lower_topics = 12, upper_topics =
90, step_size = 3)
```

Plot the relationship between number of topics and the mean pairwise cosine similarity score. Lowerbound for mean pairwise similarity is 1/x (i.e. each topic-term vector is orthogonal) where x=num_topics. After x=36 we see the similarity score doesn't drop much with an increase in num_topics. Select model with num_topics=36.




```
#Select model with num_topics=36.
N_TOPICS = 36
model_nmf = model_objs_NMF[str(N_TOPICS)]
document_topics = model_nmf.fit_transform(cv_features)
```

---

## 4. Clustering

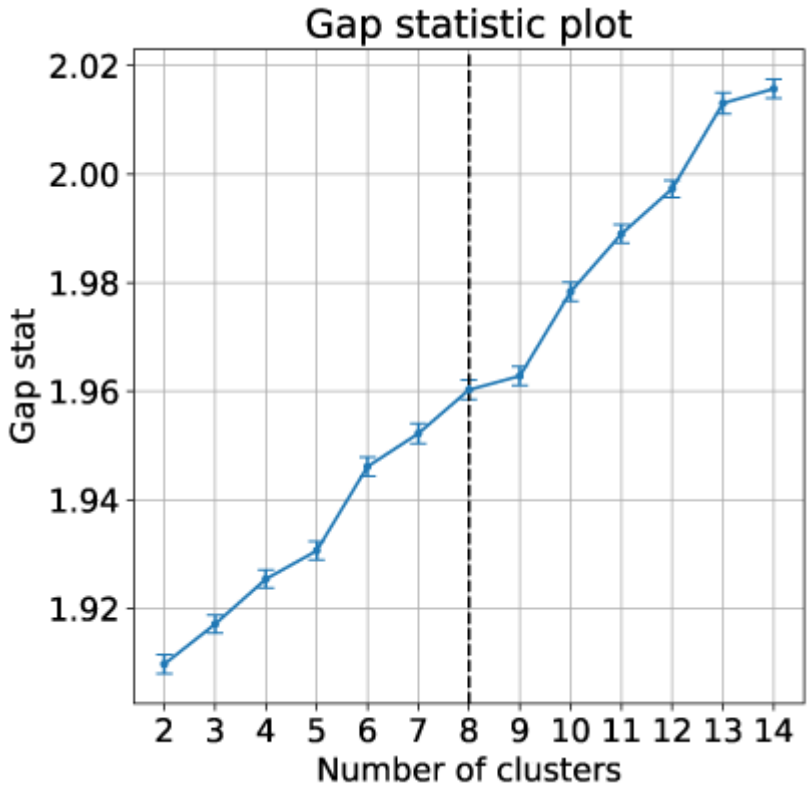
### 4.1 Get num_clusters

First we check if latent clustering structure exists for document-topic vector space, using k-means clustering.

Using the gap-statistic[4], we can say that a latent structure exists. Becuase the gap-statistic compares intra-cluster sum of squares against a reference distribution with no obvious clustering, we can say our document-topic vector space has a latent clustering structure as the gap-stat is always positive.

```
from sklearn.cluster import KMeans
from calc_k_with_gap import calc_k_with_gap

n_clusters, gaps, sks, silh_scores = calc_k_with_gap(document_topics,
max_k=16)
#next, plot the gap-statistic
```

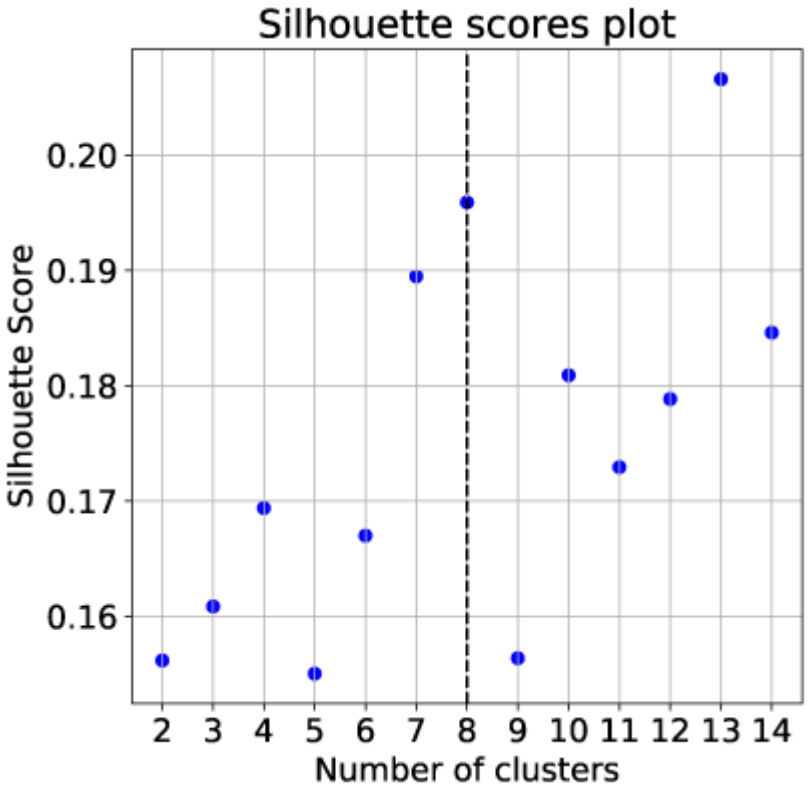



At x=8 clusters, there is an elbow - i.e. moving from x=8 to x=9 doesn't change intra-cluster distances much.

To get a better picture of whch k to choose for k-means, we use the silhouette score. The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). It ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, and the score is high, then the clustering configuration is appropriate.

```
from sklearn.metrics import silhouette_score
```

As you can see below, it gives local maxima at x=4, x=8 and x=13.




We also tried the elbow plot to help us choose. But as you can see below not much can be gleaned in terms of a distinctive elbow.


Finally, we settled on num_cluster=8 from the gap-statistic and silhouette score plots.

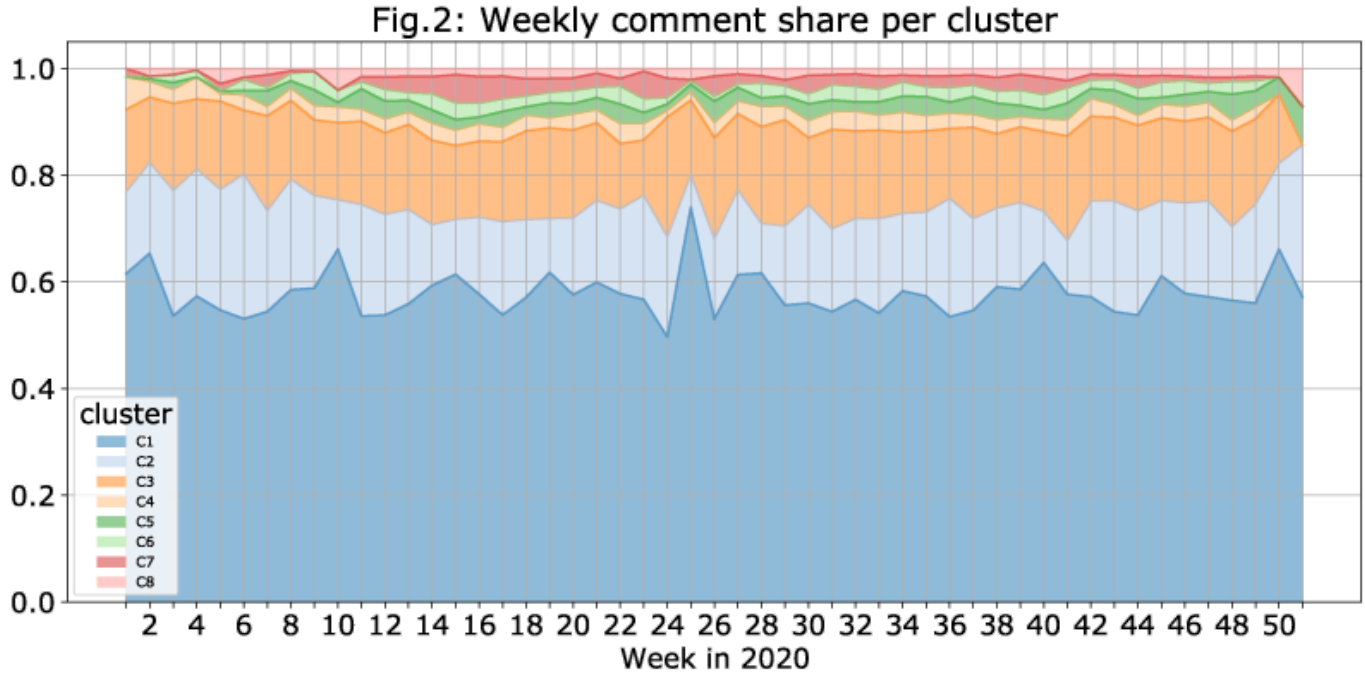
### 4.2 Get clusters

```
from sklearn.cluster import KMeans
```

```
N_CLUSTERS=8 # 10 or 8?
k_means_fit = KMeans(n_clusters=N_CLUSTERS, max_iter=300, random_state=42)
k_means_fit.fit(document_topics)
```

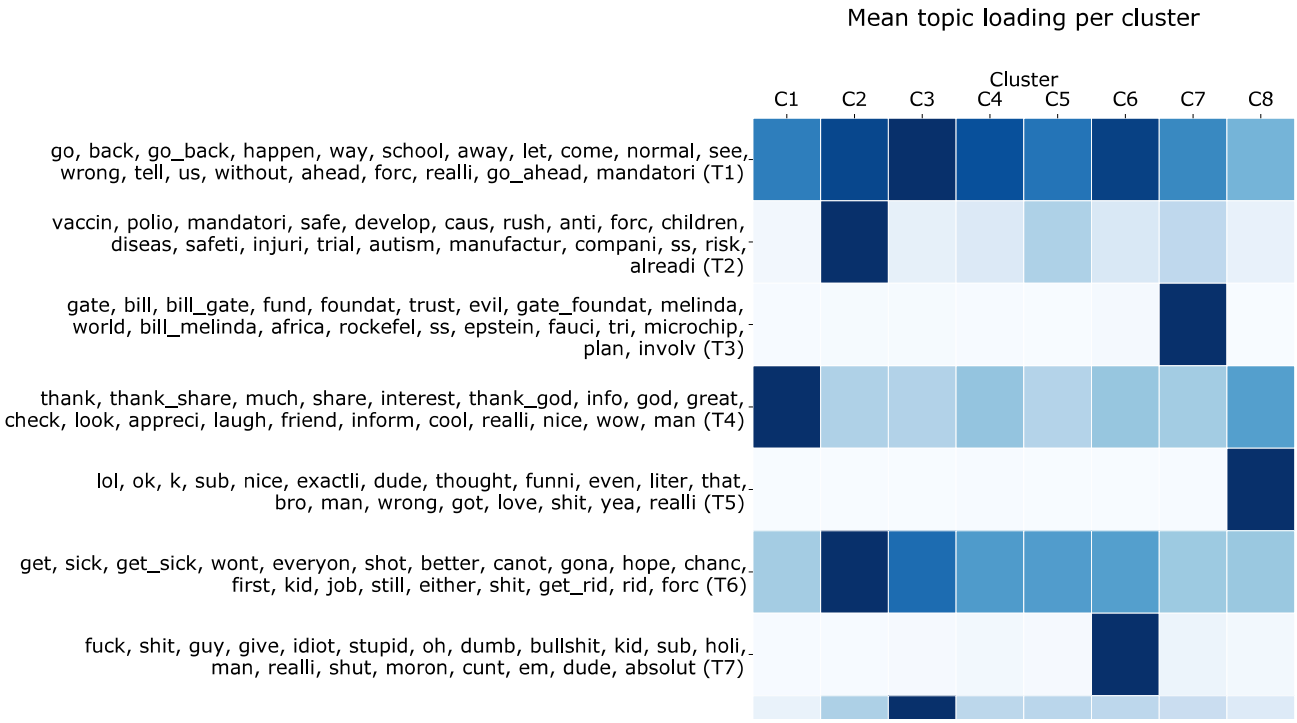
## For H2

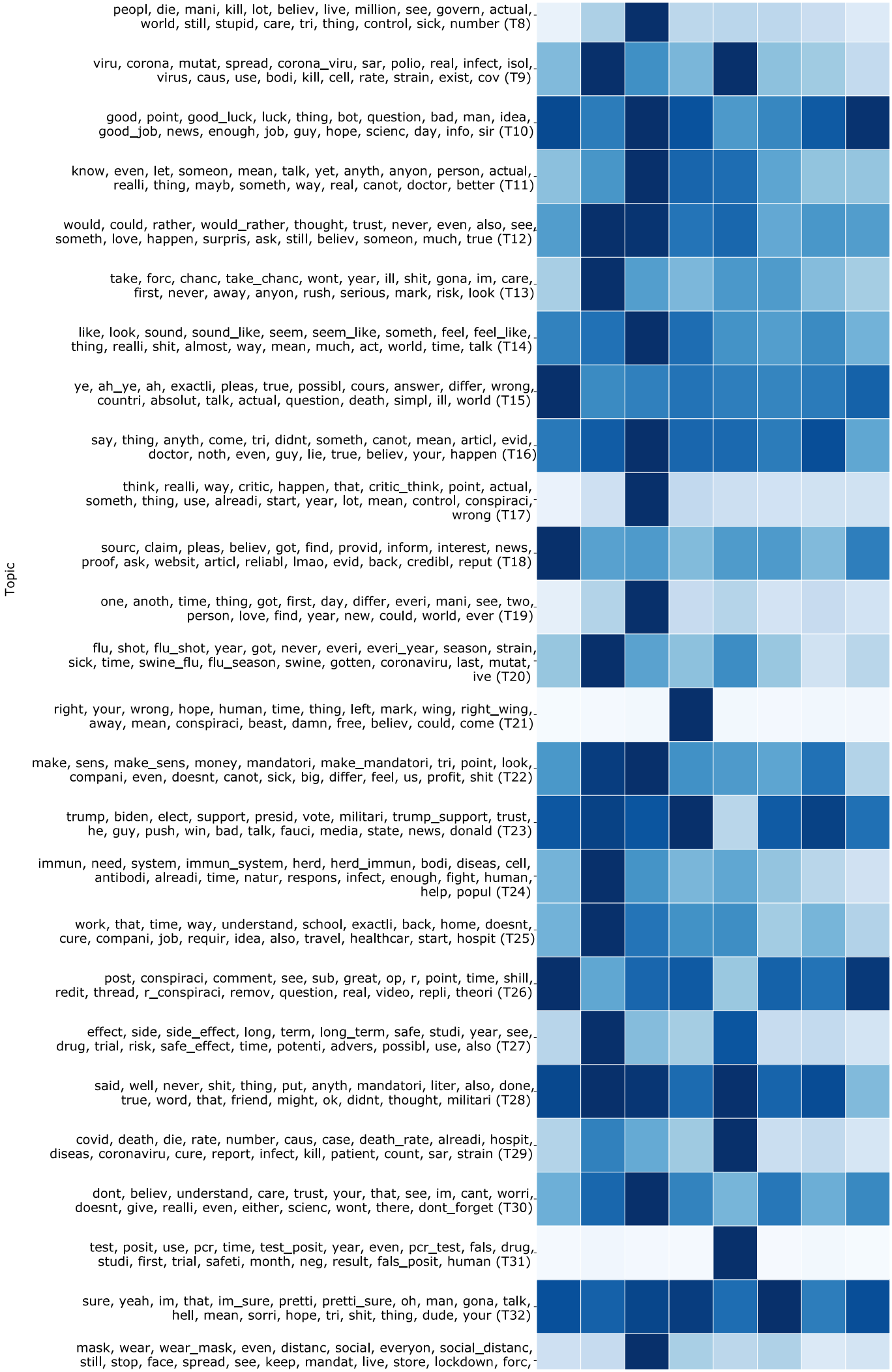
We plot the share of comments for each of the 8 clusters, per week for all weeks in 2020. The plots shows no major shifts in rhetoric.

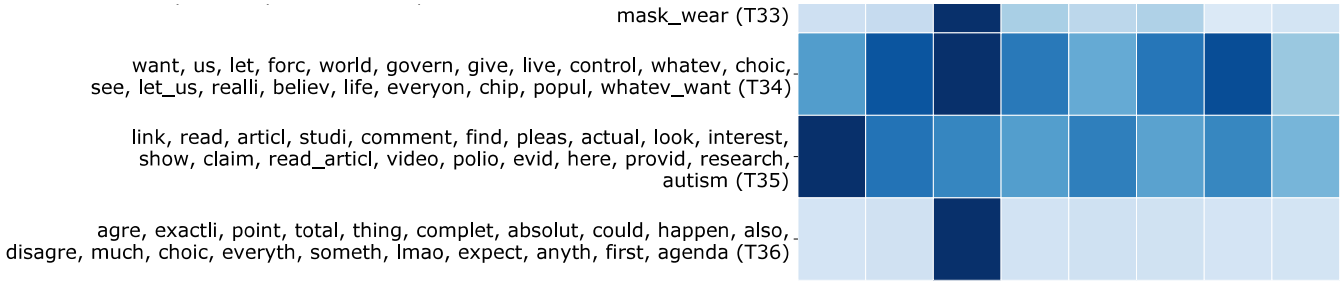



## Analysis of Topics

To interpret what each cluster was talking about, we plot the document-topic loadings averaged for each cluster.




Topic
peopl, die, mani, kill, lot, believ, live, million, see, govern, actual, world, still, stupid, care, tri, thing, control, sick, number (T8)
viru, corona, mutat, spread, corona_viru, sar, polio, real, infect, isol, virus, caus, use, bodi, kill, cell, rate, strain, exist, cov (T9)
good, point, good_luck, luck, thing, bot, question, bad, man, idea, good_job, news, enough, job, guy, hope, scienc, day, info, sir (T10)
know, even, let, someon, mean, talk, yet, anyth, anyon, person, actual, realli, thing, mayb, someth, way, real, canot, doctor, better (T11)
would, could, rather, would_rather, thought, trust, never, even, also, see, someth, love, happen, surpris, ask, still, believ, someon, much, true (T12)
take, forc, chanc, take_chanc, wont, year, ill, shit, gona, im, care, first, never, away, anyon, rush, serious, mark, risk, look (T13)
like, look, sound, sound_like, seem, seem_like, someth, feel, feel_like, thing, realli, shit, almost, way, mean, much, act, world, time, talk (T14)
ye, ah_ye, ah, exactli, pleas, true, possibl, cours, answer, differ, wrong, countri, absolut, talk, actual, question, death, simpl, ill, world (T15)
say, thing, anyth, come, tri, didnt, someth, canot, mean, articl, evid, doctor, noth, even, guy, lie, true, believ, your, happen (T16)
think, realli, way, critic, happen, that, critic_think, point, actual, someth, thing, use, alreadi, start, year, lot, mean, control, conspiraci, wrong (T17)
sourc, claim, pleas, believ, got, find, provid, inform, interest, news, proof, ask, websit, articl, reliabl, lmao, evid, back, credibl, reput (T18)
one, anoth, time, thing, got, first, day, differ, everi, mani, see, two, person, love, find, year, new, could, world, ever (T19)
flu, shot, flu_shot, year, got, never, everi, everi_year, season, strain, sick, time, swine_flu, flu_season, swine, gotten, coronaviru, last, mutat, ive (T20)
right, your, wrong, hope, human, time, thing, left, mark, wing, right_wing, away, mean, conspiraci, beast, damn, free, believ, could, come (T21)
make, sens, make_sens, money, mandatori, make_mandatori, tri, point, look, compani, even, doesnt, canot, sick, big, differ, feel, us, profit, shit (T22)
trump, biden, elect, support, presid, vote, militari, trump_support, trust, he, guy, push, win, bad, talk, fauci, media, state, news, donald (T23)
immun, need, system, immun_system, herd, herd_immun, bodi, diseas, cell, antibodi, alreadi, time, natur, respons, infect, enough, fight, human, help, popul (T24)
work, that, time, way, understand, school, exactli, back, home, doesnt, cure, compani, job, requir, idea, also, travel, healthcar, start, hospit (T25)
post, conspiraci, comment, see, sub, great, op, r, point, time, shill, redit, thread, r_conspiraci, remov, question, real, video, repli, theori (T26)
effect, side, side_effect, long, term, long_term, safe, studi, year, see, drug, trial, risk, safe_effect, time, potenti, advers, possibl, use, also (T27)
said, well, never, shit, thing, put, anyth, mandatori, liter, also, done, true, word, that, friend, might, ok, didnt, thought, militari (T28)
covid, death, die, rate, number, caus, case, death_rate, alreadi, hospit, diseas, coronaviru, cure, report, infect, kill, patient, count, sar, strain (T29)
dont, believ, understand, care, trust, your, that, see, im, cant, worri, doesnt, give, realli, even, either, scienc, wont, there, dont_forget (T30)
test, posit, use, pcr, time, test_posit, year, even, pcr_test, fals, drug, studi, first, trial, safeti, month, neg, result, fals_posit, human (T31)
sure, yeah, im, that, im_sure, pretti, pretti_sure, oh, man, gona, talk, hell, mean, sorri, hope, tri, shit, thing, dude, your (T32)
mask, wear, wear_mask, even, distanc, social, everyon, social_distanc, still, stop, face, spread, see, keep, mandat, live, store, lockdown, forc,




Using this we came up with the following labels for each cluster, based on what topics they were talking about.

| # | Label | Representative Comment |
|---|---|---|
| C1 | Skeptics | I have no idea what the truth is. What I DO know is that it isn't as it has been portrayed by the media. I'm neither confirming NOR denying the existence of COVID19. No way for me to know, or anyone for that matter. |
| C2 | Side-effects | Some vaccines and pills can work, some can kill you. Not everyone responds to medicine the same or has the same exact immune system. Humans are the guinea pigs for big pharma. And they are profit driven. |
| C3 | Political | Political ploy to get trump out of office. AOC literally tweeted this. |
| C4 | Rights | He's violating human rights, goes against Nuremberg, it's also racist to be targeting the Third World like this. |
| C5 | Safety | Semantics. It has never undergone a double-blind three phase testing process like drugs. We don't know anything about its safety or efficacy. |
| C6 | Indignant | There is ZERO fucking need for a vaccine, for a virus that 99.98% of people naturally develop antibodies for. Bill Gates is a fucking psychopath and he needs to be locked up for his crimes against humanity |
| C7 | Bill Gates | The entire plandemic is a scam to allow Bill Gates and other globalist criminals to cash in. |
| C8 | Troll | China doing something to help the rest of the world and not gain anything themselves? LOL okay. |

1. Our analysis is based on the analysis in Klein, Colin; Clutton, Peter; Polito, Vince (2018): Topic Modeling Reveals Distinct Interests within an Online Conspiracy Forum. In Frontiers in psychology 9, p. 189. ↩
2. Baumgartner, Jason; Zannettou, Savvas; Keegan, Brian; Squire, Megan; Blackburn, Jeremy (2020): The Pushshift Reddit Dataset. ↩
3. list of bots from Klein, Colin; Clutton, Peter; Dunn, Adam G. (2019): Pathways to conspiracy: The social and linguistic precursors of involvement in Reddit's conspiracy theory forum. In PloS one 14 (11) ↩

4. The Data Science Lab (2021): gap statistic – The Data Science Lab. Available online at https://datasciencelab.wordpress.com/tag/gap-statistic/, updated on 2/27/2021, checked on 2/27/2021. ↩