# Covid-19 Twitter Search

**Aditya Shetty, Jeffrey Huang, Yuanyuan Wang** and **Yuanxin Wang***
{adshetty,jchuang2,yuanyua4,yuanxinw} @cs.cmu.edu

## Abstract

Since the outbreak of COVID-19 in early 2020, people have been searching frequently on social media platforms such as Twitter about health issues. The vast amount of tweets has also made it difficult for people to quickly locate their content of interest. Using Twitter hashtags alone to label tweets still leaves an enormous corpus. In this situation, it is natural to ask the following question: how do we help people filter related tweets with more efficiency and flexibility?

We want to approach this search problem from a different perspective: instead of performing search using keywords , we find it more natural for the users to have the following user story flow: look at a few recent tweets, find the one they are more interested in, and then dig into similar contents.

To the best of our knowledge, this is an early attempt at the intersection of natural language inference, interactive data science application, Covid-19, and tweet-level search. The novelty of our project comes from this interdisciplinary idea.

**Keywords:** Covid-19, Search, Named Entity Recognition, Natural Language Inference, Sentence Embedding.

## 1 Dataset, Pipeline and EDA

### 1.1 Dataset and Data Processing Pipeline

The Dataset is obtained from the Zenodo website. It is a large-scale COVID-19 Twitter chatter dataset, which includes over 200 million tweets generated from Jan.2020 to Dec.2020. To access the data, please use the link provided in the References section.

| Dataset | Source | Time Range |
|---|---|---|
| COVID-19 Twitter | Zenodo | 01/01 - 12/06, 2020 |

Table 1: Dataset Information

The original dataset captured all languages, including English, Spanish, French, and etc. In this paper, we only focus on English tweets. Due to the limitation of computation and the compatibility with Altair and Streamlit, we randomly sampled 1000 English tweets as our test dataset.

After obtaining the sample, we used the Hydrate app to pull the original tweets from the given tweet id. Then text cleaning and pre-processing were performed on these raw tweets.

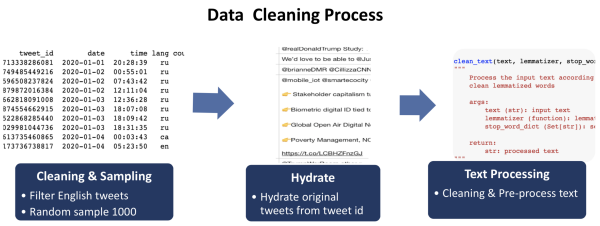


Figure 1: Data Cleaning Pipeline

### 1.2 Findings of EDA

The sampled tweets followed a bimodal distribution with peaks in May and in October. As shown in Figure 2, the tweets first increased dramatically from January to May, and then decreased gradually until September and rebounded in October. This confirms the representation of our sample since it reflected the reality of the

Language Technologies Institute, Carnegie Mellon University

spread of COVID-19. COVID-19 first became a pandemic in early spring and was spreading fast. Later the spread was slowed down for a while because of lockdowns. However, the spread got worse starting in Fall. It's normal that people generated more tweets related to COVID-19 when the spread was getting worse.

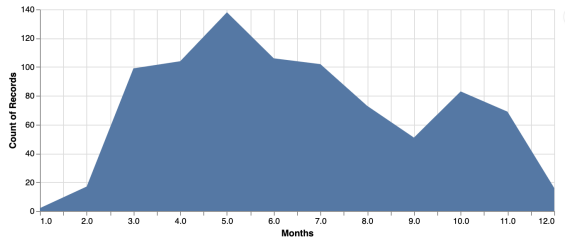


Figure 2: Distribution of the Sampled Tweets

# 2 Method

## 2.1 NER Search

Named-Entity-Recognition is the process of which document bodies extract and classify named entities in text. A named entity is a token that is rigidly associated with a type of proper noun in commonly used categories - such as names, organizations, and dates. For example, the token 2019 is rigidly associated as defining a given year in the Gregorian Calendar. Named Entity Recognition can be used to augment searches besides text alone. For example, while the University of Pittsburgh and the University of Pennsylvania may match from text similarity alone - they wouldn't necessarily match with Pomona College even though both are organizations. NER augmented search also allows for the consideration of the organization entity type in determining document similarity. The specific model we used is provided by the Spacy en_core_web_sm" model. This model is a pre-trained convolutional neural network trained on small samples of web data. The reason for this model selection was it's portability and recognizability as a baseline component - and the fact it provided NER based tokenization without creating word vectors - so no encoding takes place. This makes it an optimal baseline to the Sentence Encoder. While Spacy provides tokenization out of the box, the tweet corpus was lemmatized with the external WordNetLemmatizer and filtered to turn hashtags back into standard text. Tweets are matched based on the Jaccard Index of the sets of tokens produced by en_core_web_sm. This was evaluated by measuring cosine distances of the selected text with the question sample. Jaccard Overlap was chosen over Spacy's native NER matching method as Streamlit's computational resources prevented calculated Spacy documents on the fly or loading them from pickled objects that were produced offline. Jaccard Overlap can be done with standard sets which are portable regardless of the python environment they were created in.

## 2.2 Sentence Encoder

We use Facebook Infersent Natural Language Inference model to encode the the tweets to 4096-way vectors and then use vector level cosine similarity scoring to compute top-n similar tweets related to one tweet.

In the original Natural Language Inference task, the goal is to group the relationship between two sentences into one of three categories:entailment, neutral, or contradiction, as shown in Figure 3. In our case, however, we train the whole classification model end to end but only use the weights up to sentence encoder part to encode the tweets. For training set, we use Quora Question Pair dataset and SNLI dataset with highly relevant to daily life / common sense sentence pairs, which is similar to twitter content. We also particularly pick the Twitter Corpus version of the Glove word embedding to improve the accuracy of our model.

Since Infersent model checkpoints and Glove word embedding files are too large to load, no re-training is required and our pipeline primarily relies on offline-trained similarity score matrices. We train multiple sentence encoder models with different hyper parameters and store several similarity score matrices offline. The users can therefore tune these parameters by switching to different matrices.

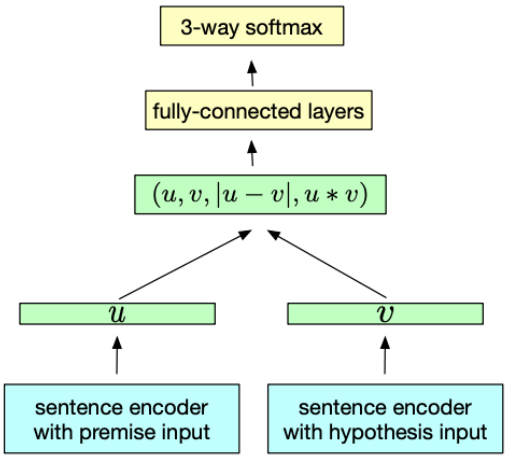


Figure 3: General Scheme for Natural Language Inference

## 3 Components

### 3.1 Visualisations

With the goal of helping users have an intuitive understanding of the text distribution in the whole corpus, we set up a word cloud visualization section.

Users can modify the word cloud by choosing a threshold frequency , filtering out stopwords and lemmatizing the corpus. Finally a hidden bar graph provides a quantitative explanation of the frequency distribution of the word cloud.

We believe this visualization component will also help the users make better sense at the NER and sentence encoder results in later sections.

Besides, one small visualizations is included to view the frequency of tweets in each month this year and the results agree with what most of us believe.

### 3.2 Interactions

We include a rich set of interactions in this project. Apart from the aforementioned word cloud, the users can also flexibly select the reference tweet they want, how many similar tweets to retrieve, and hyperparameter tuning for both NER and sentence encoder models.

### 3.3 Narrative Guidance

One extra highlight of our project is the inclusion of narrative components. We not only incorporate significant amount of explanatory knowledge descriptions to help the users better understand cutting-edge search NLP technologies, but also bring three questions in the hyperparameter tuning section. We believe that these questions will convey the idea to the users that the tuning process is not random guessing or black box; instead, why certain phenomena happen must be explained by plausible reasons and every change in the hyperparameter set must be an informed decision.

## 4 Results & Discussion

### 4.1 Human Evaluation

We evaluate our models by taking the similar tweets produced by our two models and deciding whether or not each tweet is similar to the base tweet. We then aggregate our results and perform a score comparison. In this experiment, 5 tweets are sampled, and the top 6 tweets for both the NER model and the Sentence Encoder model with learning rate $2 \times 10^{-5}$, batch size 64 and 5 training epochs are extracted and evaluated. The results are shown in Table 2.

| Tweet | NER | Sentence Encoder |
|---|---|---|
| 1261978560249683969 | 3/6 | 5/6 |
| 1249944985316794372 | 2/6 | 6/6 |
| 1295438615250472960 | 2/6 | 4/6 |
| 1256772521774518273 | 3/6 | 4/6 |
| 1260726271321018372 | 3/6 | 4/6 |
| **TOTAL** | 13/30 | 23/30 |

Table 2: A comparison of the number of produced tweets that are actually similar to the base tweet.

Just going by the tweets selectively sampled, one sees that our sentence encoder model greatly outperforms the baseline NER model in producing tweets that are actually similar to a chosen tweet. The model hyperparameter combination chosen for the sentence encoder model for this comparison is not necessarily the best combination of hyperparameters for producing similar tweets. It follows that a better set of hyperparameters can produce even better results.

While only 5 tweets are sampled for the purposes of this experiment, which leaves room for a high amount of uncertainty, one can witness that the general comparison results will hold even if sampling on a different or larger set of random tweets.

### 4.2 Hyperparameter Evaluation

While the results from before indicate a superiority of the sentence encoder model over the baseline NER model, a question still left open is figuring out which hyperparameter combination will produce the best results. One way to get closer to our desired resolution is to evaluate the influence of each hyperparameter with respect to the problem.

We execute this exploration by examining how modifying each hyperparameter individually from a default model will influence the top tweets. For this experiment, we examine how many of the top 5 tweets stay the same (and how many change) upon the changing of each hyperparameter. The results are shown in Figures 4, 5, and 6.

From the bar charts in the figures, we see that there is not a hyperparameter that is inconsequential in the computation of the top 5 similar tweets to a base tweet. However, noticing that there is a huge majority of base tweets for which the top 5 tweets are completely different between different batch sizes, we deduce that the most important hyperparameter to tune is the batch size.

One crucial point to note is that the hyperparameter combination of learning rate $1 \times 10^{-5}$, batch size 32,

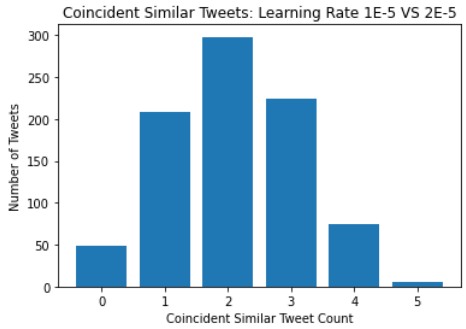


Figure 4: The number of coincident similar tweets upon changing the learning rate.

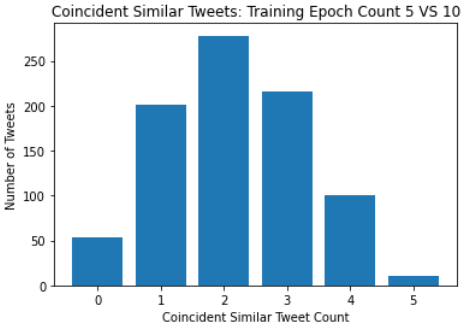


Figure 6: The number of coincident similar tweets upon changing the number of training epochs.

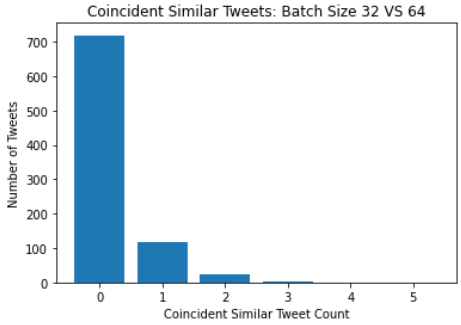


Figure 5: The number of coincident similar tweets upon changing the batch size.

and training epoch count 10 produces a matrix for which several tweets do not have themselves as the most similar tweet. An implication is that the computation for the similarity score has the potential to result in undesired variance. This means that we would like for the learning rate to be high enough for scores to sufficiently reflect similarity, consider larger (but not too large) sizes of batches, and train the model on the data enough but not too much to make the similarity scores less deviant.

## 5 Future Work

As discussed, one of the major bottlenecks in this project is the resource limit of Streamlit web application. Even for the offline similarity matrices, we need limit the number of testing examples in order to avoid exceeding git-lfs limit. In the future, it would be interesting to deploy this application on a GPU server where the application is expected to make online predictions for new input with more flexibility. In this way, we can include 10x test data, the complete model checkpoints, as well as different word embedding.

Another area of future work is on the user interface part. Due to the limit of time, we only find a few pictures for the existing reference tweets to improve the UI. In the future, the user interface can be enhanced with front-end code (e.g., React) which automatically retrieve twitter image using APIs.

## References

Banda, Juan M. (2020). *"A Twitter Dataset of 40+ million tweets related to COVID-19"* Zenodo. URL https://zenodo.org/record/4308491#.X9HRCxNKh0s

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP),pages 670–680, Copenhagen, Denmark,September 2017. Association for Computational Linguistics.