WHAT REMAINS OF THE APOCALYPSE

ANALYZING DISASTER TWEETS, FOR PATTERNS AND MORE

Figure Source

Haizhou Liu, Di Lin and Li Zhou

PART A. INTRODUCTION

Natural disasters are known as sudden, unpredicted and destructive events, that cause significant losses to the environment and human society. Not much is left for analysis after the disasters, except for severe utility damage and high death/injury tolls; however, social media platforms such as Twitter would leave us with the legacy of numerous social interaction records (e.g. tweets), that remind us of what was happening at that very moment. Such disaster-related tweets typically include warnings and precaution advice before the disaster, rescue and donation efforts afterwards, as well as mental support throughout. With state-of-the-art data mining tools, we could better identify the nature of such thousands of tweets, meanwhile looking for interesting patterns and trends in the Twittersphere that co-occurred with the disasters.

In this blog post, we demonstrate our data analysis process and list of findings on disaster-related tweets collected by CrisisNLP, using Python and data-mining packages such as Pandas, Scikit-Learn and TensorFlow. In **PART B**, we provide a detailed description of the tweet dataset, together with how we further cleaned it and retrieved more supplementary information. In **PART C**, we analyze the statistical properties of the tweet corpus, including tweet trends, lexical/semantic measures, as well as tweeting similarities among disasters. In **PART D**, we analyze the Twitter networks around the disasters. In **PART E** and **PART F**, we respectively perform supervised learning and unsupervised learning on the dataset, while analyzing the word embeddings generated as a by-product of the training process. We conclude our findings in **PART G**. All python codes for data analysis can be found on this GitHub repository (please contact us if you cannot access it).

PART B. TWEET RETRIEVAL AND PREPROCESSING

In 2021, Alam et al. published a disaster-related tweet dataset entitled *Human-Annotated Disaster Incidents Data from Twitter* (HumAID). This dataset contains a total of 77,196 tweets collected on 19 worldwide disasters during 2016-2019, covering earthquakes, hurricanes, wildfires and floods. The dataset is open-access on CrisisNLP, where the tweets are grouped by disaster, split into train/dev/test sets, and therefore stored in 57 different CSVs. Each of the CSV file contains three columns: tweet id, tweet text and label; the label column identifies the category of the tweet content (prayer and support, rescue and donation efforts, etc.).

For the ease of study, we use the Pandas package to merge the data files together for the train/dev/test CSVs respectively, with three additional columns specifying the year, location and type of the disaster. Also, realizing that the current features of this dataset is not sufficient for a well-rounded research, we further employ the Twitter API and the Tweepy package to scrape for supplementary information of these tweets (e.g. time created, user info and like/retweet count), which significantly enhances the richness of features. Considering there are a few cases where we fail to retrieve the tweets, we add a “Response-Code” column to indicate the type of error in tweet retrieval (e.g. User suspended and status not found). With all the above steps, we finally yield 3 large dataframes (namely the train, dev and test dataframe), with all informative columns enumerated in Table 1.

Table 1 Informative columns of the final dataset and their brief description

Column Name	Description	Data Type
id_str	Tweet ID in the form of string	string
tweet_text	Content of the tweet	string
class_label	Category of the tweet content (provided by HumAID)	string
place	Location of the disaster (Country, District or City)	string
disaster	Type of disaster	string
year	Year of the disaster	int
created_at	Time the tweet was created	datetime
entities	Hashtags and other contents embedded in the tweet	json
favorite_count	Number of favorites of this tweet	int
retweet_count	Number of retweets of this tweet	int
user	User Information	json
Response-Code	Error Message of Tweet Retrieval (or “Successful”)	string

PART C. STATISTICAL PROPERTIES OF TWEETS

First of all, we are intrigued in the statistical properties of the disaster-related tweets. For example, tweeting trends show how the tweeters keep their interest in the disaster; the use of hashtags show what topics the tweeters are interested in; various text measures (e.g. reading difficulty and sentiment score) show the variations in user content. Therefore, in this part, we intend to track both the trends of general properties, as well as interesting patterns in the various text measures. For most of this section, we focus entirely on one particular disaster: Hurricane Harvey, as it contains the largest number of tweets for statistical analysis. In section C3 we also discuss the similarities and differences between different disasters.

C1. GENERAL TRENDS

We first create a stacked bar chart in Fig. 1 to visualize the tweeting trends of different categories of tweets. It can be seen that the tweets shifted from cautions/advice and damage reports before August 29, to rescue/volunteering/donation efforts and mental support afterwards. In general, the number of tweets was mild before August 29, but surged drastically since then, when Harvey was reported to be discontinuing.

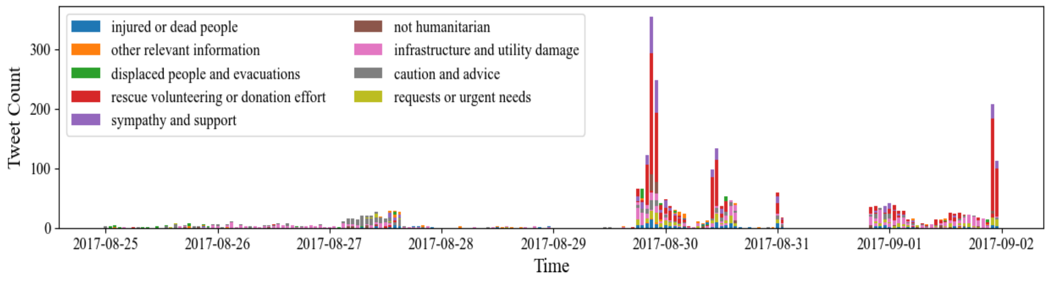


Fig. 1 Trends of different categories of Harvey-related tweets.

We also plot the mostly used hashtags among these tweets in Fig. 2, in which an exponential-like decay is observed for the usage statistics of hashtags. Among these 30 hashtags we have identified four major categories: (a) Just tagging Harvey (e.g. #Harvey and #Hurricane2017); (b) Reporting News (e.g. #Breaking and #News); (c) Prayer and Support (e.g. #Houstonstrong and #HarveyRelief); and (d) Reflections (e.g. #Climatechange and #Trump). We can see that

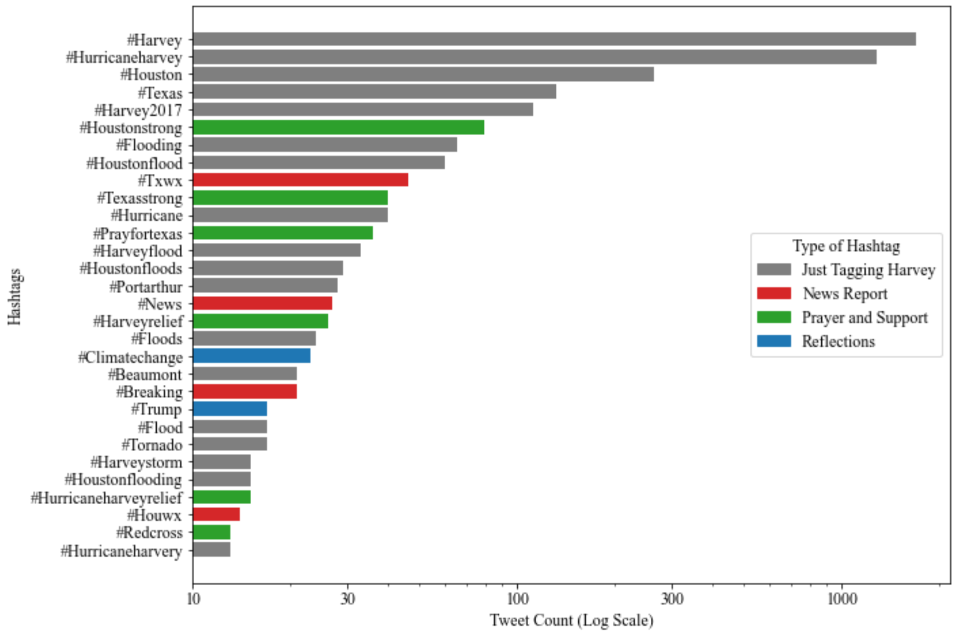


Fig. 2 Mostly used hashtags of Harvey-related tweets.

the mostly used hashtags are trivial in that only the Harvey disaster is mentioned. Prayer/Support hashtags are the next popular, while reflection-like hashtags are hardly observed. In Fig. 3 which demonstrates the usage ratio of different types of hashtags with respect to time, we can see that news-related hashtags occurred frequently only before August 29, while prayers and support lasted throughout the period.

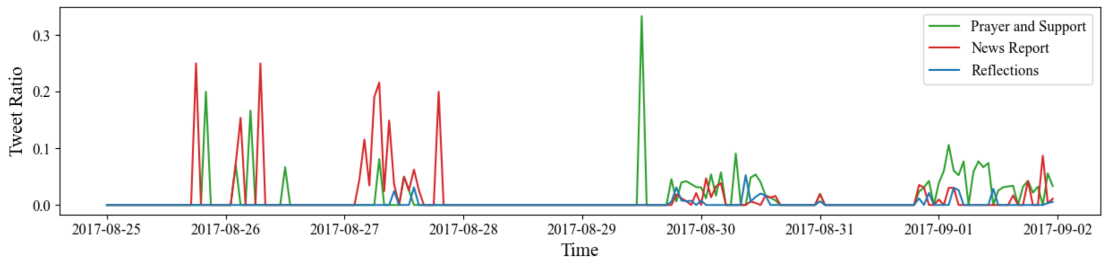


Fig. 3 The ratio different types of hashtags were used with respect to time.

Finally, we compare the most popular words in the tweets before and after August 29, which is clearly a watershed for the tweet corpus. Keywords "Harvey" and "Hurricane" are removed before two separate word clouds are respectively created (Fig. 4). Before and during the hurricane, tweets were focusing mostly on floods, damages and victims; cry for help, support efforts and prayers became the mainstream topic after Harvey.

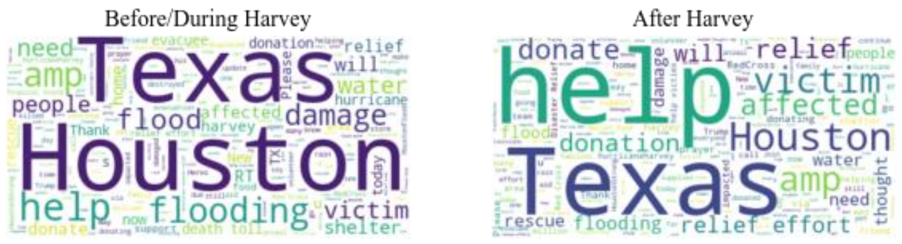


Fig. 4 The ratio different types of hashtags were used with respect to time.

C2. LEXILE/SEMANTIC TEXT MEASURES

To further dig into the tweets, we employ the following lexile/semantic measures to evaluate the structure of these tweets:

(1) Number of hashtags employed (using the *entities* attribute of the retrieved json);
(2) Number of favorites (using the *favorite_count* attribute of the retrieved json);
(3) Flesch-Kincaid reading score;
(4) Dale-Chall reading score;
(5) Number of sentences;
(6) Number of lexicons;
(7) Number of syllables;
(8) Sentiment score (ranging from -1 to 1).

(1) and (2) are directly obtained from the json we scraped before. (3)-(7) are obtained by calling the *textstat* package. (8) is obtained by calling the *vaderSentiment* package.

Fig. 5 visualizes the distribution of these measure statistics. It can be seen that most of the measures follow either a skewed normal distribution or an exponential decay; interestingly, for semantic scores, most tweets are neutral, but there are also a significant portion of tweets with sentiment on the positive/negative extreme.

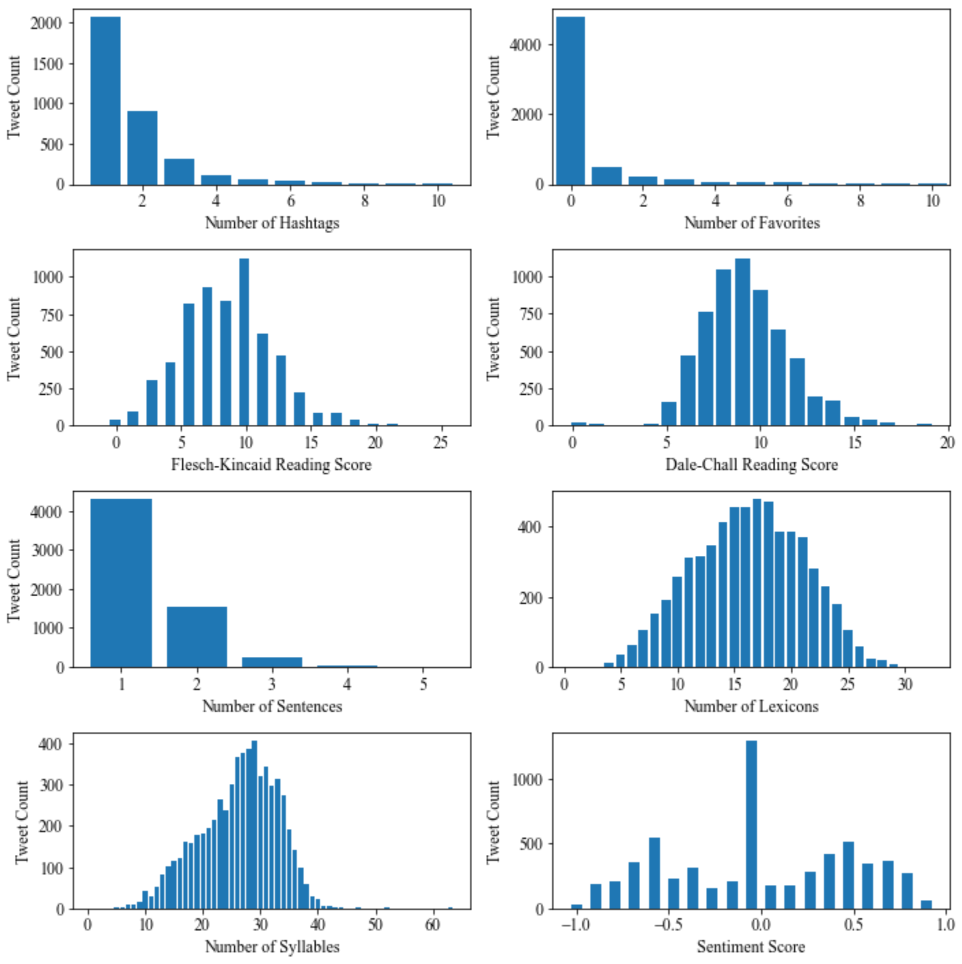


Fig. 5 Distribution of the lexile/semantic text measures of Harvey-related tweets.

Figure 6 shows the correlation of these text measures using a heatmap. We can see that the measures are largely independent towards each other; however, the two reading scores are strongly correlated, so are the syllable/lexicon/sentence counts.

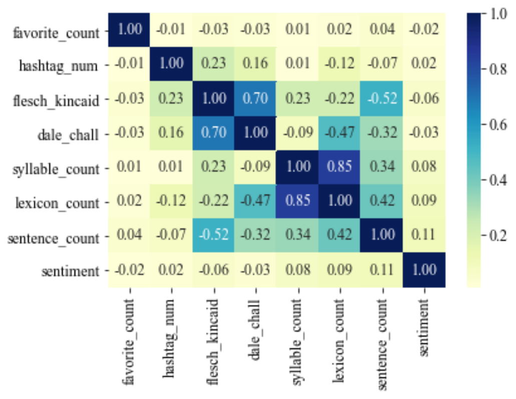


Fig. 6 Correlation between the text measures of Harvey-related tweets.

To explore the trends of these measures with respect to time, we group and average the text measures of tweets in each hour, and perform seasonal decomposition to see the trends, seasonal components and residuals. Fig. 7 shows the decomposition plot for (normalized) sentiment scores, where we can see a seasonal cycle of ~1 day, and the trends first went negative then slowly climbed towards positive. This should be understandable, since the tweeters should be at first devastated at the news, but then gained hope as rescues and donation efforts were in place. Please note that the visualization is interactive so that readers can select the measure of their interest; see this GIF for demonstration.

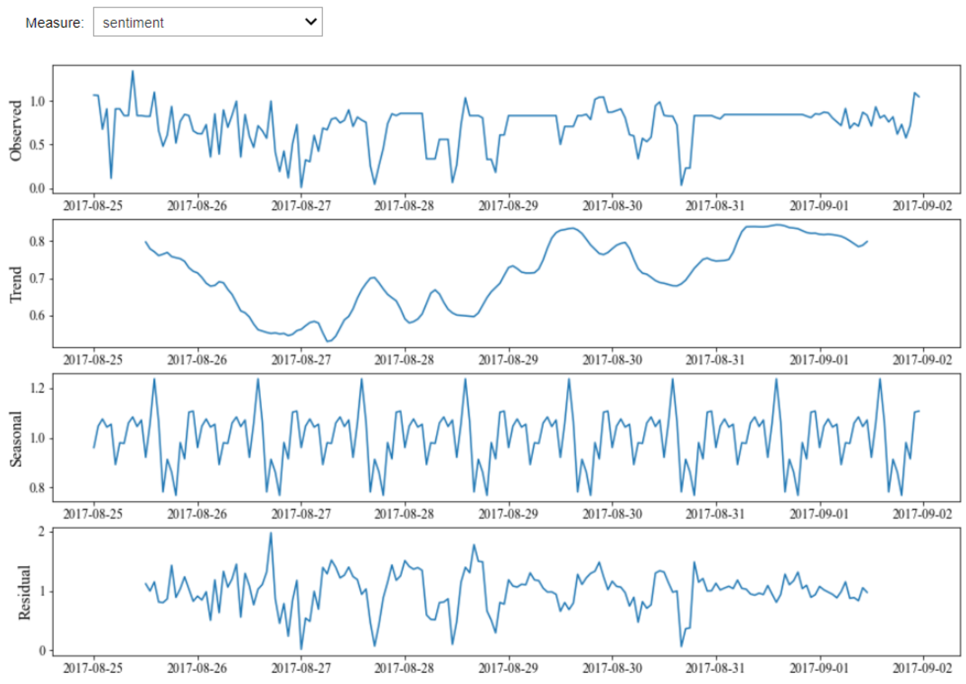


Fig. 7 Seasonal decomposition of the average sentiment score across time.

C3. SIMILARITY OF TWEETING PATTERNS ACROSS DISASTERS

By far we have only discussed the tweeting patterns of Hurricane Harvey. But are the patterns alike among different types of disasters? To check, we also plot a stacked bar chart of the tweeting trends on the 2016 earthquake that struck Kaikoura, New Zealand (Fig. 8). Not surprisingly, the trends are dissimilar to that of Hurricane Harvey, with most tweets exploding on November 14, the day just after the earthquake, but no tweet beforehand. This is

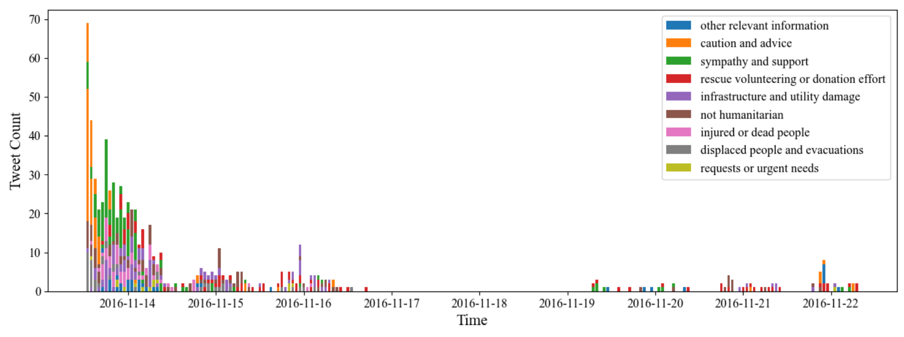


Fig. 8 Trends of different categories of Kaikoura-Earthquake-related tweets.

plausible since earthquakes are more impossible to predict than hurricanes, leading to fewer warnings before the incident and more emotional shocks afterwards.

A question arises as to how the tweeting trends among disasters are similar to each other; here we employ Dynamic Time Warping (DTW), a similarity measure for time-series that calculates “edit distances” as metric. Here we choose two pairs of disasters for DTW demonstration (Fig. 9): (1) Kaikoura Earthquake V.S. Puebla Earthquake (left); and (2) Kaikoura Earthquake V.S. Hurricane Harvey (right). It is fairly obvious that the tweeting pattern is very similar for the two earthquakes, with a final edit distance of 662; whereas for the earthquake-hurricane pair, it’s hard to find a match between the two time-series (as shown in the gray dashed lines attempting to match the curves), with a large edit distance of 6,378. We are tempted to conclude from the results that, disasters of the same type might share a similar tweeting trend.

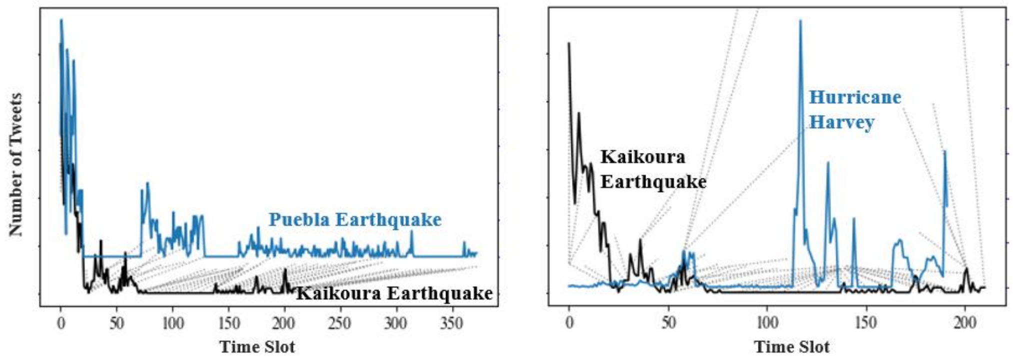


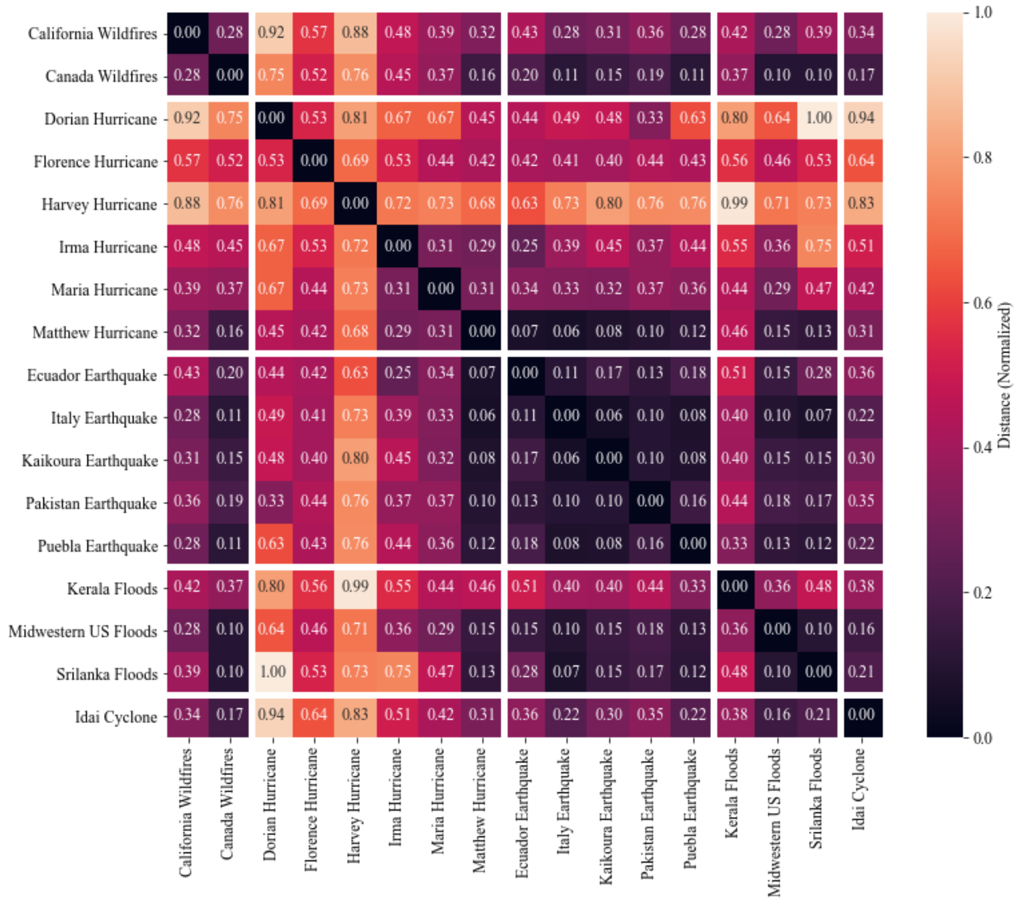
Fig. 9 Dynamic Time Warping between the tweeting trends of: Kaikoura Earthquake and Puebla Earthquake (left), as well as Kaikoura Earthquake and Hurricane Harvey (right). Dashed Gray lines indicate DTW matching between two curves.

We further created a heatmap (Fig. 10) to visualize the normalized DTW distance between every two disaster. To help us quickly extract information, the heatmap has been split into small “blocks” of different disaster types. We can see that disasters of the same type indeed lead to a greater trend similarity, especially for earthquakes, floods and wildfires; on the other hand, there are also cases where different disaster types can yield similar trends, e.g. earthquake and wildfires.

PART D. SOCIAL NETWORKS AROUND THE DISASTERS

Another important topic in social media is the social network emerging around them, which we explored in this part using NetworkX. Two types of networks are analyzed: user-mention networks (i.e. “@user” embedded in tweets) and retweet networks. Again, we focus entirely on Harvey-based tweets, and all tweets unsuccessfully retrieved are eliminated from the corpus.

D1. USER-MENTION NETWORKS



	California Wildfires	Canada Wildfires	Dorian Hurricane	Florence Hurricane	Harvey Hurricane	Irma Hurricane	Maria Hurricane	Matthew Hurricane	Ecuador Earthquake	Italy Earthquake	Kaikoura Earthquake	Pakistan Earthquake	Puebla Earthquake	Kerala Floods	Midwestern US Floods	Srilanka Floods	Idai Cyclone
California Wildfires	0.00	0.28	0.92	0.57	0.88	0.48	0.39	0.32	0.43	0.28	0.31	0.36	0.28	0.42	0.28	0.39	0.34
Canada Wildfires	0.28	0.00	0.75	0.52	0.76	0.45	0.37	0.16	0.20	0.11	0.15	0.19	0.11	0.37	0.10	0.10	0.17
Dorian Hurricane	0.92	0.75	0.00	0.53	0.81	0.67	0.67	0.45	0.44	0.49	0.48	0.33	0.63	0.80	0.64	1.00	0.94
Florence Hurricane	0.57	0.52	0.53	0.00	0.69	0.53	0.44	0.42	0.42	0.41	0.40	0.44	0.43	0.56	0.46	0.53	0.64
Harvey Hurricane	0.88	0.76	0.81	0.69	0.00	0.72	0.73	0.68	0.63	0.73	0.80	0.76	0.76	0.99	0.71	0.73	0.83
Irma Hurricane	0.48	0.45	0.67	0.53	0.72	0.00	0.31	0.29	0.25	0.39	0.45	0.37	0.44	0.55	0.36	0.75	0.51
Maria Hurricane	0.39	0.37	0.67	0.44	0.73	0.31	0.00	0.31	0.34	0.33	0.32	0.37	0.36	0.44	0.29	0.47	0.42
Matthew Hurricane	0.32	0.16	0.45	0.42	0.68	0.29	0.31	0.00	0.07	0.06	0.08	0.10	0.12	0.46	0.15	0.13	0.31
Ecuador Earthquake	0.43	0.20	0.44	0.42	0.63	0.25	0.34	0.07	0.00	0.11	0.17	0.13	0.18	0.51	0.15	0.28	0.36
Italy Earthquake	0.28	0.11	0.49	0.41	0.73	0.39	0.33	0.06	0.11	0.00	0.06	0.10	0.08	0.40	0.10	0.07	0.22
Kaikoura Earthquake	0.31	0.15	0.48	0.40	0.80	0.45	0.32	0.08	0.17	0.06	0.00	0.10	0.08	0.40	0.15	0.15	0.30
Pakistan Earthquake	0.36	0.19	0.33	0.44	0.76	0.37	0.37	0.10	0.13	0.10	0.10	0.00	0.16	0.44	0.18	0.17	0.35
Puebla Earthquake	0.28	0.11	0.63	0.43	0.76	0.44	0.36	0.12	0.18	0.08	0.08	0.16	0.00	0.33	0.13	0.12	0.22
Kerala Floods	0.42	0.37	0.80	0.56	0.99	0.55	0.44	0.46	0.51	0.40	0.40	0.44	0.33	0.00	0.36	0.48	0.38
Midwestern US Floods	0.28	0.10	0.64	0.46	0.71	0.36	0.29	0.15	0.15	0.10	0.15	0.18	0.13	0.36	0.00	0.10	0.16
Srilanka Floods	0.39	0.10	1.00	0.53	0.73	0.75	0.47	0.13	0.28	0.07	0.15	0.17	0.12	0.48	0.10	0.00	0.21
Idai Cyclone	0.34	0.17	0.94	0.64	0.83	0.51	0.42	0.31	0.36	0.22	0.30	0.35	0.22	0.38	0.16	0.21	0.00

Fig. 10 Normalized DTW distance between every two disasters.

To create a user-mention network, we first make the following assumptions:

(1) All relationships are mutual and weighted (i.e. weighted and directed graph). It's not the best model, but it's what we can do with few data;
(2) If User A mentions User B, then there is a mutual relationship between A and B;
(3) If User A mentions both User B and User C, then there is an additional mutual relationship between B and C;
(4) If there are *n* interactions between users A and B, then their mutual weight is set to *n*.

A weighted and undirected graph is therefore created, the edge plot of which is visualized in Fig. 11 using a spring layout. Much to our surprise, only a small subset of users are connected to each other (in the center); other users tend to form small networks among themselves.

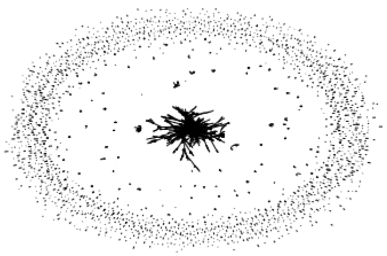


Fig. 11 Edge graph of the user-mention network among Harvey-related tweets.

The degree distribution follows the power law as expected, with the majority of users holding a degree of only one. The top 20 users with the largest degrees have been plotted in Fig. 12. Humanitarian organizations such as @RedCross, political entities such as @realDonaldTrump and @Potus, as well as news media such as @ABC and @CNN, gain the most user mentions. It is interesting to note that these users happen to be the ones with the highest centrality measures in whatever centrality metric.

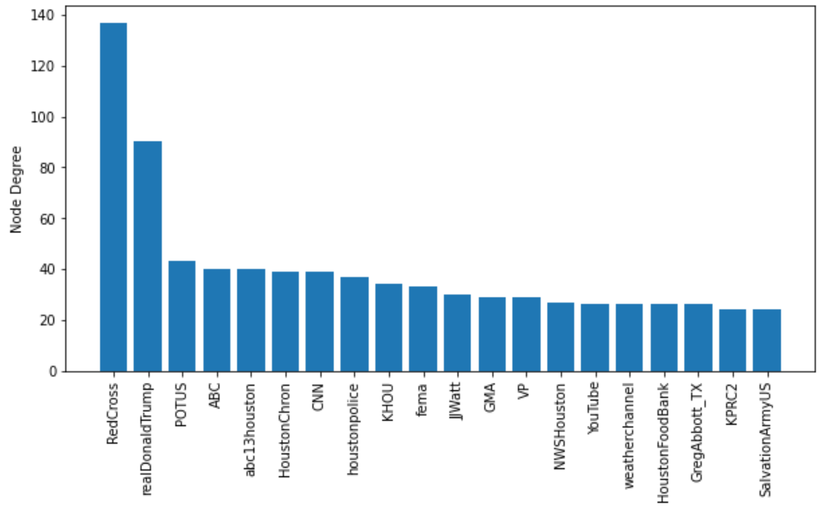


Fig. 12 Twitter accounts that had the most degrees in the Harvey-related user-mention network.

As mentioned before, the user-mention network is sparsely related, so we are interested in exploring patterns in its connected components (1,823 components in total!!). The largest component, not surprisingly, lies in the central cluster of Fig. 11, which is mostly a radial network. Fig. 13 shows some interesting patterns in other connected components: Component #2 exhibits a tree-like structure with chains of user-mentions; Component #4 exhibits a highly radial network, where the user account in the center @LindaSN0228WI

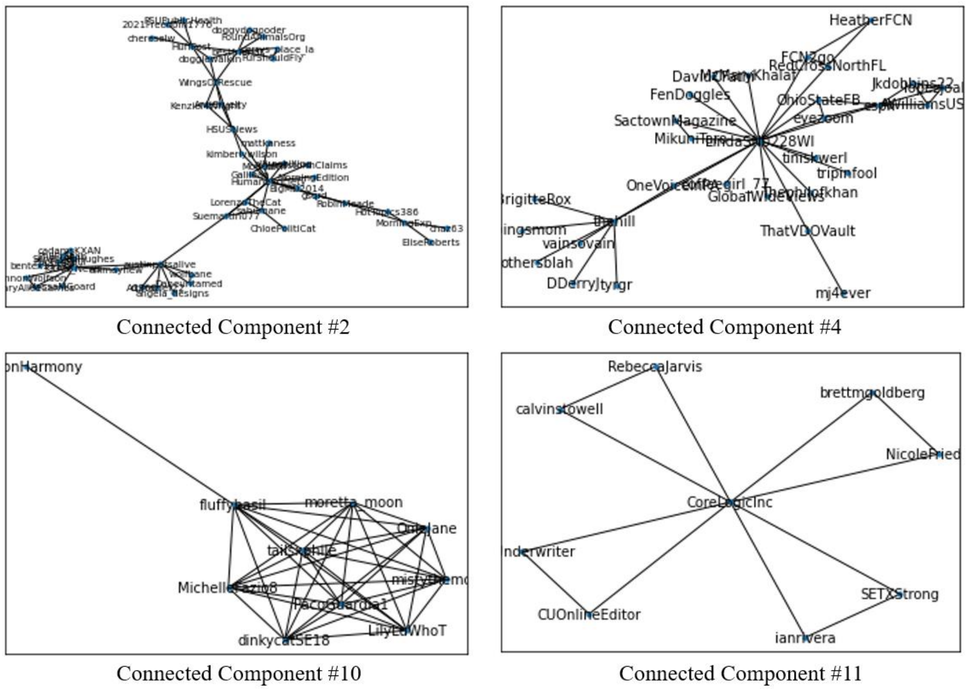


Fig. 13 Some connected components of the Harvey-related user-mention networks.

happens to be a freelance writer and photographer; Components #10 exhibits an almost strongly-connected network; Component # 11 exhibits a triadic network where @CoreLogicInc is a financial services company that offers financial help amidst the disaster.

D2. RETWEETING NETWORKS

The retweeting network is expected to be fundamentally different from the user-mention network. Intuitively it should be a directed network. and with the low cost of retweeting, users might repost anyone's tweets, whether they know the person in reality or not, leading to a randomly-connected and very complicated network. This is confirmed by the network graph (Fig. 14), where it is almost impossible to differentiate nodes that are detached from the large web of retweets as the retweeting behavior is largely stochastic. But in fact, there are even more connected components in this (2,011 as compare to 1,823 in the user-mention network), indicating this network is deceptively dense due to the many long edges across the spring layout.

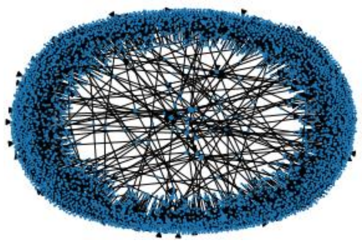


Fig. 14 Network graph of the retweeting network among Harvey-related tweets

Slightly different than the user-mention network, users with the highest degree centrality in the retweeting network are almost exclusively news media and freelance reporters, which provide reliable and breaking news sources to the tweeters. The connected components of the retweeting network are also more uniform in size, in that there are no dominant components that significantly outweigh others (recall the large cluster back in the user-mention network). Even the largest component contains only 37 nodes. To further demonstrate, Fig. 15 plots the number of nodes in the top 20 connected components, from which we can find a smooth decrease in component size.

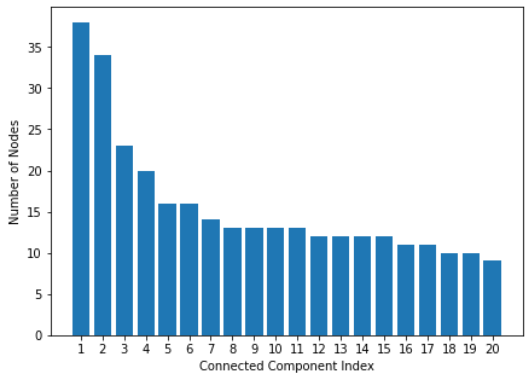


Fig. 15 Number of nodes in the top 20 connected components of the Harvey-related retweeting network.

PART E. SUPERVISED LEARNING: PREDICTING TWEET CATEGORIES

One of the primary intents of the HumAID dataset is to correctly classify the disaster tweets by their contents, to match the manually-labeled categories (which can be found in the *class_label* column). To get a sense of what the "categories" refer to, we hereby enumerate all the 10 unique labels below:

(1) Displaced people and evacuations;	(2) Sympathy and support;
(3) Requests or urgent needs;	(4) Rescue volunteering or donation effort;
(5) Infrastructure and utility damage;	(6) Injured or dead people;
(7) Caution and advice;	(8) Not humanitarian;
(9) Missing or found people;	(10) Other relevant information.

Considering the nature of the classification task, we decide to discard some of the supervised learning including decision trees and fully-connected neural networks, which cannot fully exploit the series datatype and tends to be incompetent for large numbers of features (say 1,000 features after TF-IDF vectorization). We finally land on three supervised learning techniques: logistic regression, Bi-LSTM as well as BERT, which are either efficient or believed to work well with natural languages.

E1. LOGISTIC REGRESSION

Before performing logistic regression, we first lemmatize all words in the document, and apply the TF-IDF vectorizer in scikit-learn to vectorize the documents into vectors of 1,000. In the vectorization process, we set both single words and bigrams as potential features, while discarding words that appeared more than 90% of the time. Then, a standard logistic regression classifier is created by once again calling scikit-learn, after which the TF-IDF vectors and the class labels are respectively passed in as inputs and outputs.

Surprisingly enough, the simple logistic regression algorithm performs fairly well on the tweet dataset. It reaches an accuracy of 73.4%, a precision of 72.9%, a recall of 65.8% and an F-1 score of 68.2%, which are even comparable to state-of-the-art language models including BERT.

We take a peek at the topic words which are most important to the prediction of each label. This is reflected in the magnitude of the fitted linear coefficients for each word. Fig. 16 shows the word importance in determining each topic (there is also an interactive version where you can select the topic you are interested in, and see the coefficient chart for only that topic. You can find the demonstration here). It can be seen that only a few words serve as dominant features for determining each topic (presented as tall spikes). These few words, however, are extremely relevant to the topics: for example, in the "infrastructure and utility damage" category, the most salient words are "damag", "destroi" and "destruct", whose importance to identifying this category are self-explanatory.

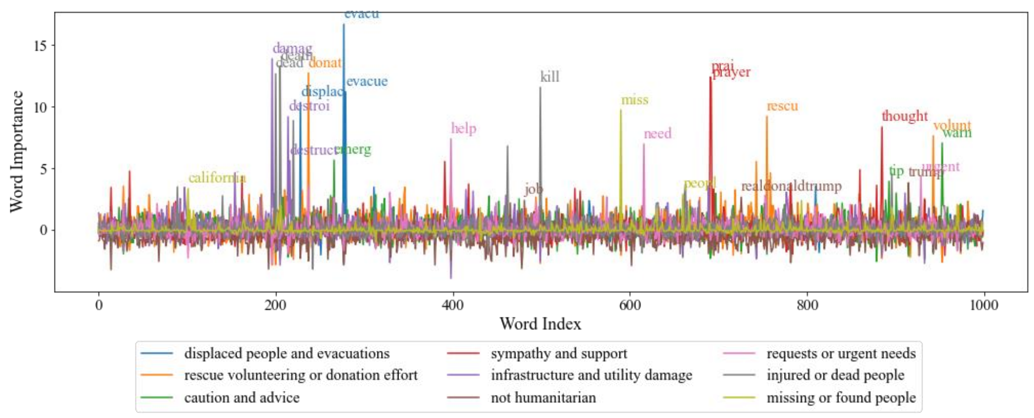
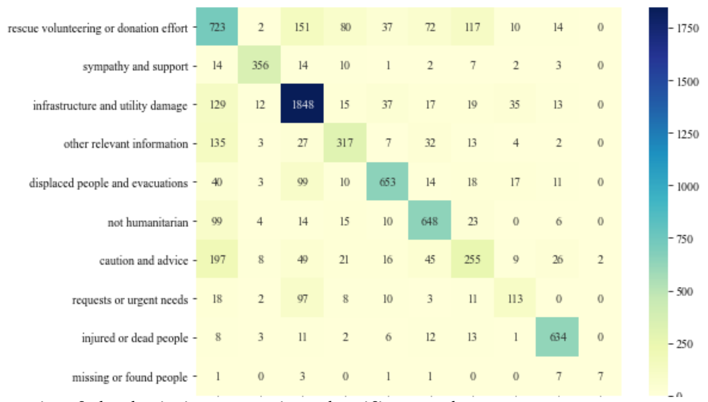


Fig. 16 Importance of the feature words to determining each topic.

As a side note, we also explore whether the tweets can be traced back to disasters with logistic regression, i.e. use the *disaster* column as labels. Apparently, this kind of prediction is much easier, with an accuracy of 96.3% and an F-1 score of 95.8%. The most likely explanation is that most users would tag or at least mention the disaster when tweeting.

Finally, we are interested in how fair the model is in predicting different categories. The confusion matrix generated on the test set has been plotted in the form of a heatmap in Fig. 17, where we can see the model is more biased towards majority groups such as "injured or dead people", while against minority groups such as "caution and advice". This is an unideal situation, especially when we need to classify tweets mostly containing minority group messages. In order to construct a fairer model for all categories, we call the imblearn package to oversample the training set, so that every category contains the same number of tweets in the oversampled training set. Fig. 18 shows that the accuracy has now become more "balanced" across categories, although the overall accuracy performance has been somewhat compromised.



rescue volunteering or donation effort	723	2	151	80	37	72	117	10	14	0
sympathy and support	14	356	14	10	1	2	7	2	3	0
infrastructure and utility damage	129	12	1848	15	37	17	19	35	13	0
other relevant information	135	3	27	317	7	32	13	4	2	0
displaced people and evacuations	40	3	99	10	653	14	18	17	11	0
not humanitarian	99	4	14	15	10	648	23	0	6	0
caution and advice	197	8	49	21	16	45	255	9	26	2
requests or urgent needs	18	2	97	8	10	3	11	113	0	0
injured or dead people	8	3	11	2	6	12	13	1	634	0
missing or found people	1	0	3	0	1	1	0	0	7	7

Fig. 17 Confusion matrix of the logistic regression classifier on the test set.

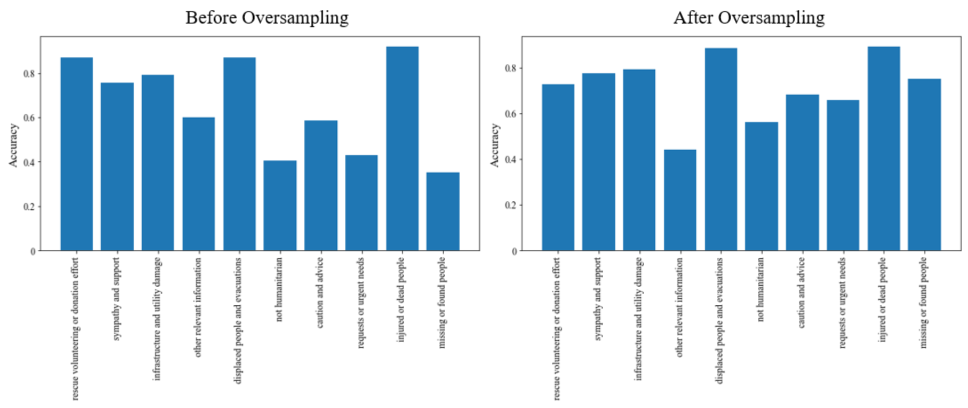


Fig. 18 Accuracy of the logistic regression classifier on different categories, before and after oversampling.

E2. BI-LSTM

Logistic regression, despite being highly efficient, does not leverage the nature of natural language as a continuing sequence. Bi-LSTM, as an extension of Long-Short Term Memory (LSTM), underscores the sequence of words by order of appearance, and uses gate logics to smartly “forget” or “remember” the features of words that preceded or succeeded the current word. Each word goes through an LSTM block with identical parameters, so that the training burden is condensed to only a small block of trainable parameters.

To conduct text preprocessing, model construction and model fitting conveniently and coherently, we call the Tensorflow 2.0 library and its Keras backend to finish these tasks. We execute our codes on Google Colab Pro to avoid out-of-RAM errors. We also reference this text classification tutorial for coding details.

First in text preprocessing, we perform the following steps in order:

(1) Build an empty tokenizer, with 5,000 word slots for tokenization;
(2) The tokenizer is fit on the (lemmatized) training texts to finalize on the words of choice. Words not chosen are replaced with <OOV>.
(3) Transform all the words in the original text sentences to their indices;
(4) Pad the transformed sentences to align with the longest sentence with zeros.

After preprocessing, the training set is now a matrix of size (Number of samples, Maximum number of words in a sentence), where each entry is an index of the current word. The training set is then passed onto the constructed Bi-LSTM model, which consists of an embedding layer, a Bi-LSTM layer, and a fully-connected layer for prediction. The schematic of this neural network architecture is shown in Fig. 19 (references this article).

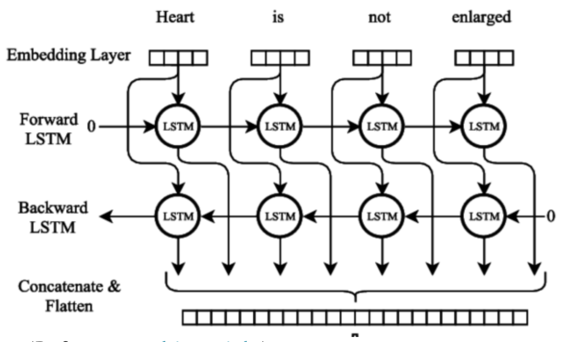


Fig. 19 Bi-LSTM architecture (References this article).

We then fit the Bi-LSTM to the transformed training set and the class labels. The fitting process converged quickly with only 2 epochs, reaching a classification accuracy of 74.6% and a categorical loss of 0.74 on the test set. It therefore outperforms logistic regression slightly by 1.2 percentage points. Fig. 20 presents the change of accuracy/loss with respect to training epochs.

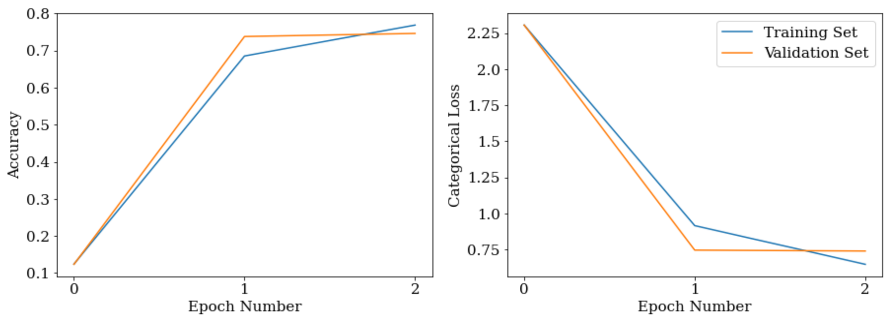


Fig. 20 Changes of accuracy/loss with respect to training epochs.

One important by-product of the Bi-LSTM model training process is its fitted embedding layer, a matrix that stores the representation vector for every word recorded by the tokenizer. With the embedding layer, we can infer the similarity of words by comparing the similarity of their representation vectors. Fig. 21 shows a 2D t-SNE plot on the embeddings of the most frequent words. Words are highlighted in color if they are also topic words for a particular

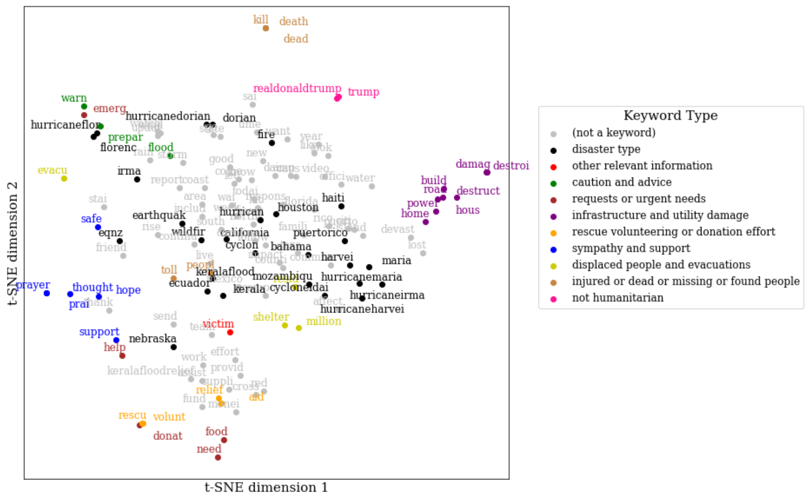


Fig. 20 Two-dimensional t-SNE plot of the word embeddings for high-frequency words.

category back in logistic regression. Most words of the same category are aggregated together, which shows the effectiveness of the embedding layer.

It is noteworthy that LSTM alone could not handle this task properly. The training set accuracy gets stuck at 28.1%, regardless of the number of epochs. It tells us that for disaster-related tweets, backward word relations are as important as forward ones.

E3. BERT

With the gaining popularity of transformers in machine learning, Bidirectional Encoder Representations from Transformers (BERT) is gradually becoming one of the go-to solutions for advanced natural language processing. Different from the above Bi-LSTM model which only requires a sequence of single words as inputs, BERT requires token embeddings, segment embeddings as well as position embeddings. The 3 types of inputs are then passed through the transformer encoder which is extremely complicated in architecture, and are transformed into vectors of a given shape. Finally, the vectors are passed through a dropout layer and a fully-connected layer for classification.

For our task, we choose the most complicated L=12, H=768 model (also known as BERT-base) for the best classification performance. A simplified diagram on the model architecture has been sketched in Fig. 21 (Note that the complicated transformer encoder is not elaborated. We refer to the BERT tutorial on TensorFlow for preprocessing text and fine-tuning BERT.

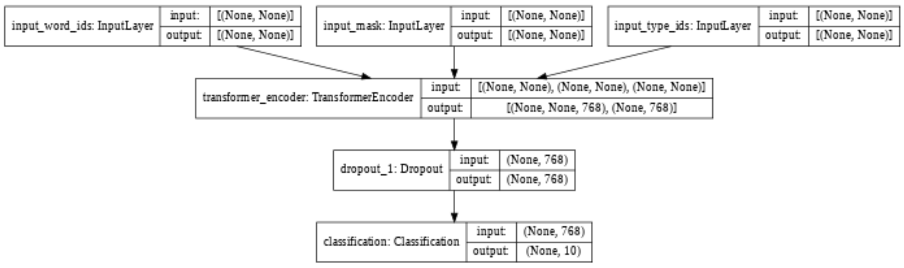


Fig. 21 BERT model architecture (simplified).

As the BERT encodings would significantly consume RAM and memory, we stick to Google Colab Pro for the entire analysis, and fix the batch size to be 25 even in the final prediction step. Sparse categorical cross entropy is set as the loss function; sparse categorical accuracy and F-1 score is set as the evaluation metric. The learning rate is set to be 2×10^{-5}.

With only one epoch of fine-tuning (takes ~7 hours with TPU!!), the classification accuracy has risen from 19% to 77.3%, and the F-1 score has also increased from 5% to 72.7%. This remarkable performance is attributable to the fact that the pretrained BERT model already captures the main features of natural language, and therefore preserves the patterns of some natural language features. To demonstrate, we have listed out the word embeddings in the

BERT model before and after training. Words of the same category already tend to cluster before training, and surprisingly little improvements can be observed afterwards. We speculated that with the word embeddings already close to optimal with pre-training, the fine-tuning process mostly shifts its focus on improving the final fully-connected layer.

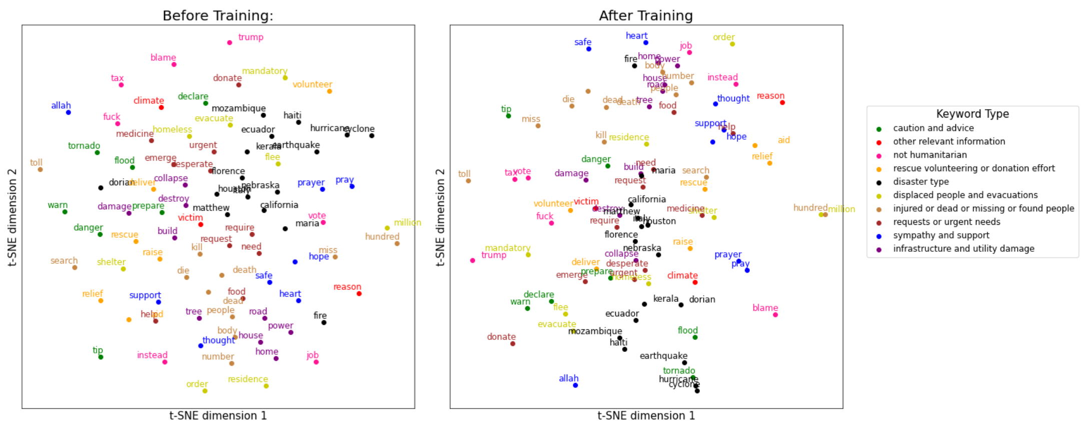


Fig. 22 Word embeddings before and after BERT fine-tuning.

To recap on the supervised learning part, we list the accuracy score, F-1 score and time-consumption for the 3 types of classifiers in Table 2. We can see that generally there is a trade-off between accuracy/F-1 score and time consumption/required computing power, with logistic regression being the most efficient and BERT being the most accurate. In real practices we need to carefully consider our requirements before selecting an appropriate learning method.

Table 2 Performance of different methods on classifying the disaster-related tweets.

Method	Accuracy	F-1 Score	Running Environment	Computation Time
Log-Reg	73.4%	68.2%	Local CPU	~30 seconds
Bi-LSTM	74.6%	70.3%	Google Colab Pro (GPU)	~15 minutes
BERT	77.3%	72.7%	Google Colab Pro (TPU)	~ 7 hours

E4. TRANSFOMERS

Due to the superior performance of BERT model, we further analyzed the performance of different model constructs in the Transformers family. Since our task is a typical sequence classification problem, we utilized the following pretrained Bert variants from huggingface which are good at such tasks.

Table 3 Performance of different Transformers on classifying the disaster-related tweets.

Pretrained Model	Model ID	# Parameters	Accuracy	Computation Time

BERT	bert-base-cased	109M	76.7%	~ 3 hours
ALBERT	albert-base-v2	11M	76.7%	~ 2 hours
RoBERTa	roberta-base	125M	76.8%	~ 3.5 hours
DistlBERT	distilbert-base-uncased	66M	77.1%	~ 1.5 hours
FlauBERT	flaubert/flaubert_base_cased	54M	65.4%	~ 3.5 hours
Funnel Transformer	funnel-transformer/small-base	115M	76.7%	~ 3 hours

All models were trained on Google's Colab Pro platform for 3 epochs with the same hyper parameters (batch size, learning rate, weight decay, etc.). Except for FlauBERT, all transformer models achieved similar accuracy scores around 77%. It is interesting to note that DistlBERT, despite having a smaller model size, still performed comparatively well within a much shorter training time. By analyzing the trend of training and validation losses, all of the models exhibited signs of overfitting after training for 3 epochs. Therefore, for our dataset it is probably sufficient to train for 3 epochs or less.

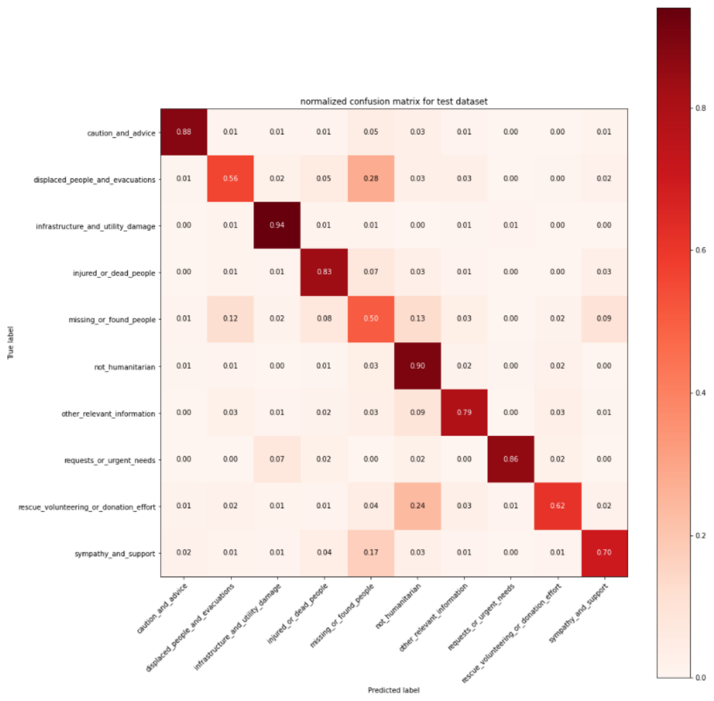


Fig. 23 Normalized confusion matrix of the transformer classifier on the test set.

By examining the normalized confusion matrix of our model predictions, the class "Missing or Found People" was predicted with least accuracy. There could be two reasons. Firstly

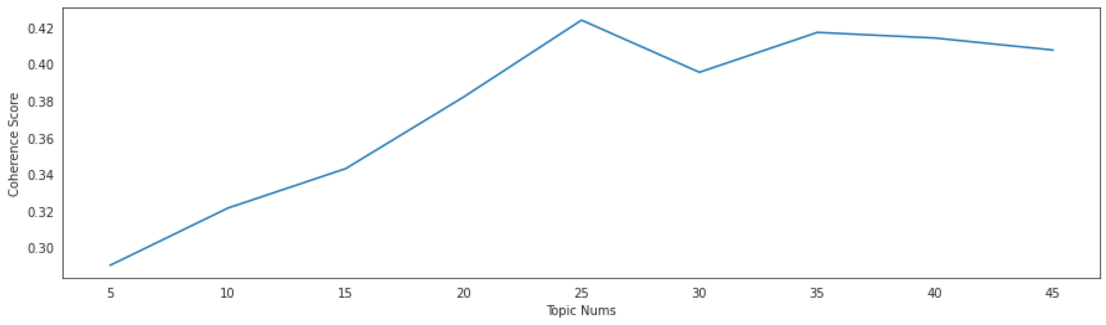
“Missing or Found People” has the smallest sample size among all classes. Since sampling was done randomly during training, having a smaller training size might affect the model’s ability to learning from this class. Secondly, this class and the other class (“Displaced People and Evacuations”) are quite ambiguous in context for the model to distinguish as they share similar words and phrases. Therefore, in subsequent studies we could try to improve the model performance by oversampling the “Missing or Found People” class or combine the two classes “Missing or Found People” and “Displaced People and Evacuations” as one.

PART F. UNSUPERVISED LEARNING: TOPIC MODELING ON TWEET TEXT

In unsupervised learning part, we used Non-Negative Matrix Factorization (NMF) and Latent Dirichlet Allocation (LDA) to analyze our tweet text.

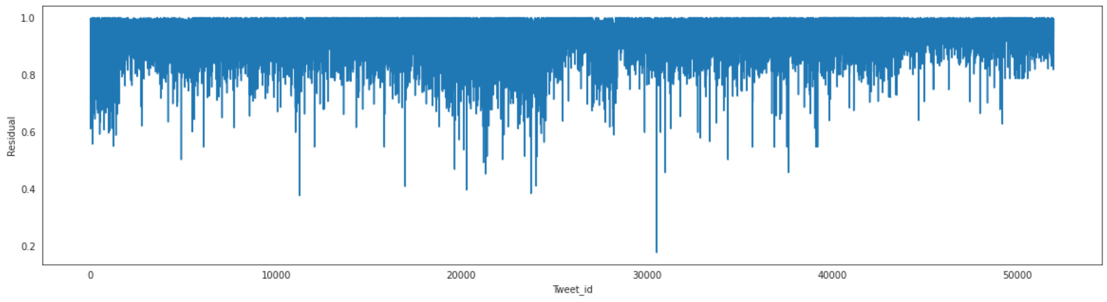
First, we are interested in how many topics are there in those tweet text, thus we used NMF to calculate the coherence score for different numbers of topics. Its core part logic is similar with clustering, which is to ‘cluster’ the data points into multiple clusters. If the coherence score is high, it means this number of topics is making more sense than others.

We used Nmf and CoherenceModel from Genism package and used get_coherence api to calculate the coherence score.

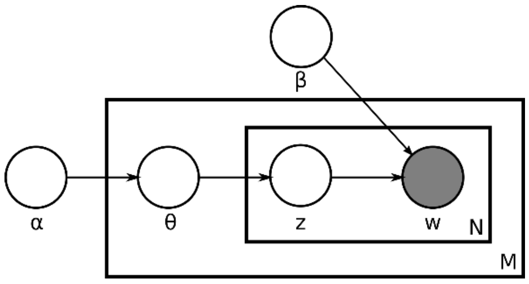


From the graph above, we can see there are 25 topics between 2016 to 2019 which we can guess there may around 25 events (there is high coherence score for 35 as well) occurred in these 4 years.

As we know 25 is the optimal number for topics, we manually insert this number into NMF model. Then we calculate the residual for each tweet to indicate whether it is highly related to the topic. If the residue scores close to 0, it means this tweet is highly related to a topic, otherwise, it is not. The basic logic of this method is to calculate the distance between each tweet with the NMF. Components_ and select the smallest number as its residual.

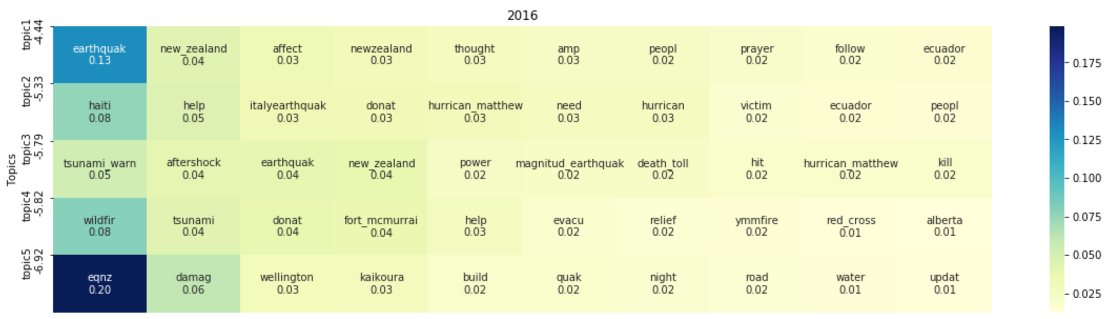


Besides NMF, we used Latent Dirichlet Allocation (LDA) to find topic in tweet text from year 2016 to year 2019. LDA is a topic discovery algorithm wildly used in unsupervised machine learning.

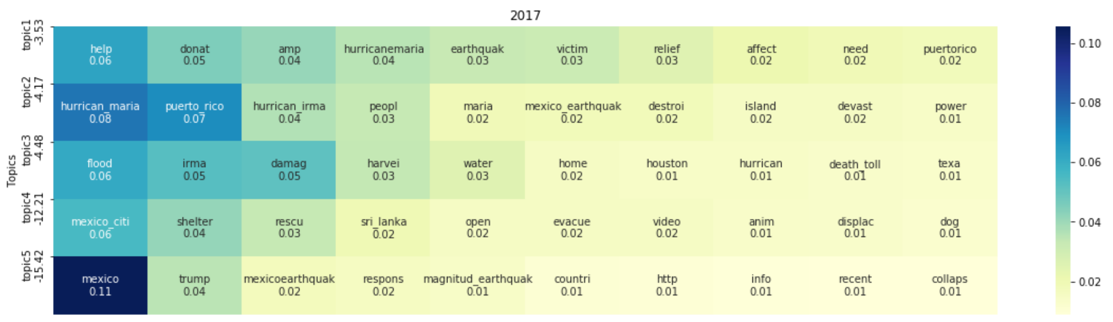


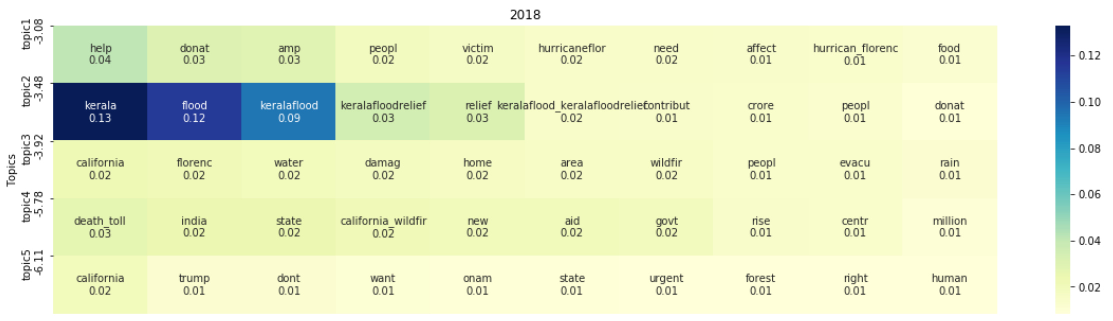
With plate notation, we can capture the dependencies of these variables concisely. The outer plate is documents, and the inner plate is the repeated word positions in a certain document. M denotes the number of documents; N denotes the number of words in a document. α is the parameter of the Dirichlet prior on the per-document topic distributions. Β is the parameter of the Dirichlet prior on the per-topic word distribution. Theta is the topic distribution for document and W is the specific word.

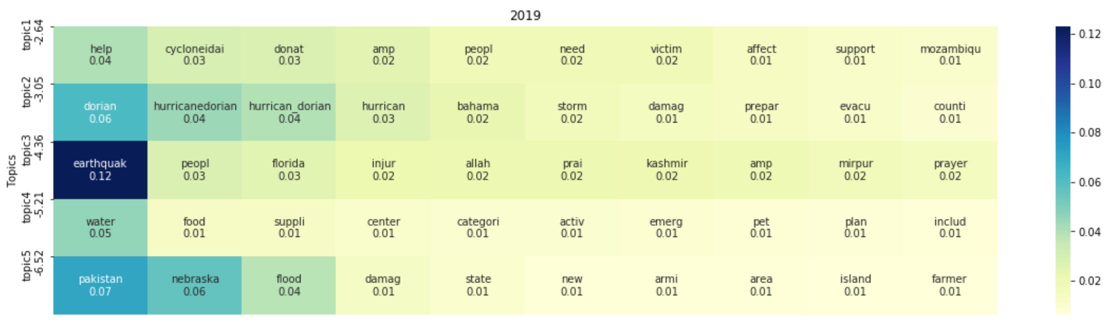
In our practice, we tokenized the words from the original text and did regular preprocessing on the tokens including removing punctuation, lower and lemmatizing. Then we add bigrams into tokens as many phrases are commonly appeared as 2 words. At last, we split the data into years and fit our unigram and bigrams into LDA model.



topic1 -4.44	earthquak 0.13	new_zealand 0.04	affect 0.03	newzealand 0.03	thought 0.03	amp 0.03	peopl 0.02	prayer 0.02	follow 0.02	ecuador 0.02
topic2 -5.33	haiti 0.08	help 0.05	italyearthquak 0.03	donat 0.03	hurrican_matthew 0.03	need 0.03	hurrican 0.03	victim 0.02	ecuador 0.02	peopl 0.02
topic3 -5.79	tsunami_warn 0.05	aftershock 0.04	earthquak 0.04	new_zealand 0.04	power 0.02	magnitud_earthquak 0.02	death_toll 0.02	hit 0.02	hurrican_matthew 0.02	kill 0.02
topic4 -5.82	wildfir 0.08	tsunami 0.04	donat 0.04	fort_mcmurrai 0.04	help 0.03	evacu 0.02	relief 0.02	ymmfire 0.02	red_cross 0.01	alberta 0.01
topic5 -6.92	eqnz 0.20	damag 0.06	wellington 0.03	kaikoura 0.03	build 0.02	quak 0.02	night 0.02	road 0.02	water 0.01	updat 0.01







By utilizing this approach we can cluster the most popular topics in a given year, helping us understand what happened in that year(What disaster came in that year, where did the disaster happened), and what did user react to the disaster. From the table above, we can realize there are multiple disaster type in 2016 including earthquake, tsunami and wildfire, especially the Haiti earthquake triggered the biggest discussion on Twitter. In 2017, we can see the hurricane maria in Puerto Rico and Earthquake in Mexico City dominated the year. In 2018, Kerela floods and California wildfire occupies the top2 topics, while Hurricane Dorian and flood in Pakistan led the 2019. Through out the 4 consecutive years, we found earthquake and wildfire continuously catches people eyes and raises the most discussion on Twitter.

Besides above, we found 'help' is the single word which contributed the most appearances. Thus people do use Twitter as a platform to seek help or share resources to help others, which confirmed Twitter's value as a tool for social good and the potential to expand its capability to make more influence in future.

PART G. CONCLUSION

In conclusion, we have used multiple data mining tools, including trend analysis, lexile/semantic measures, time series similarity analysis, network analysis and supervised/unsupervised learning, to find patterns of disaster-related tweets. Some of the major findings are listed below:

(1) The trends of disaster tweets shift from cautions/advice and damage reports, to rescue/volunteering/donation efforts and mental support. The topic words also experienced the same kind of shifting.
(2) Most lexical measures of tweets follow either an exponential decay or a skewed normal distribution, while their sentiments are clustered on both the neutral region and the very extremes;
(3) Similarities are high for similar types of disasters, but low for different types;
(4) For the user-mention network, only a small subset of users are connected to each other (in the center); other users tend to form small networks among themselves;
(5) The retweeting network is more stochastic, although it is still sparsely connected;
(6) Generally there is a trade-off between evaluation metrics and time consumption/required computing power, with logistic regression being the most efficient (~30 seconds) and BERT being the most accurate (77.3% accuracy);
(7) Using LDA model, we found earthquake and wildfire were the most discussed disaster topic on Twitter.

We hope our work, focusing on multiple facets of disaster tweets, can shed some light on disasters of the new era, as well as the social network interwoven around them.

As our final deliverable to the MADS program, we would also like to give our special thanks to all the MADS faculty and staff, who prepared such wonderful courses and reached out so timely and kindly in times of need. We hope you would be pleased when you finished reading our project, knowing that all your hard work at nurturing next-generation data scientists has finally paid off. Thank you.