Natural Language Processing (NLP) with Disaster tweets

Sina Abbasi
University of Saskatchewan
Canada
sia942@usask.ca

ABSTRACT

Twitter is one of the most popular social media these days, and the best place to get breaking news instantly. Monitoring Twitter is interesting for a lot of companies, e.g. disaster relief organizations. In this regard, we are going to build a model for the disaster tweets classification task of whether the tweets are pointing to a real disaster or not. To overcome this problem, we proposed a model using contextualized BERT embedding layer. Our proposed model shows performance improvement in comparison to the base model with LSTM structure in terms of F1 score. The source code can be found here: https://git.cs.usask.ca/sia942/disastertweetsclassification

KEYWORDS

Natural Language Processing, BERT, text classification

ACM Reference Format:
Sina Abbasi. 2022. Natural Language Processing (NLP) with Disaster tweets. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Nowadays, thanks to smartphones and easy access to the internet across the world, free social media applications are one of the best places to get breaking news instantly. Especially after COVID-19 hit, news consumption from social media rapidly increased, where based on a Pew Research Center survey conducted 31 August - 7 September 2020, about half of US adults (53%) get their news" often" or" sometimes" from social media; for example, about half of Twitter's users get their news from there as you can see in Fig. 1. [8]. Moreover, people use Twitter to report emergency news that they see in real-time. Therefore, many companies and news agencies are interested in monitoring Twitter to get real-time and breaking news. Besides, organizations such as disaster relief organizations want to know the news about disasters as soon as possible. This monitoring cannot be done by humans because we are dealing with on average 6000 [4] tweets per second.

Dataset and task in this project are taken from the Kaggle competition "Natural Language Processing with disaster tweets" [2]. The dataset contains 10000 tweets from Twitter that have been extracted by some keywords, e.g., ablaze, accident, death, bleeding, etc. Dataset is human-labeled in the way that which tweets explain a real disaster or not. Subsequently, the task for this project is defined to be classifying tweets based on real and non-real disaster tweets. Pragmatically monitoring tweets by only analyzing the words in each sentence separately is not a promising approach. For example, the author of a tweet said: "On plus side LOOK AT THE SKY LAST NIGHT IT WAS ABLAZE" [3]; the word "ABLAZE" is used not in actual meaning but metaphorically.

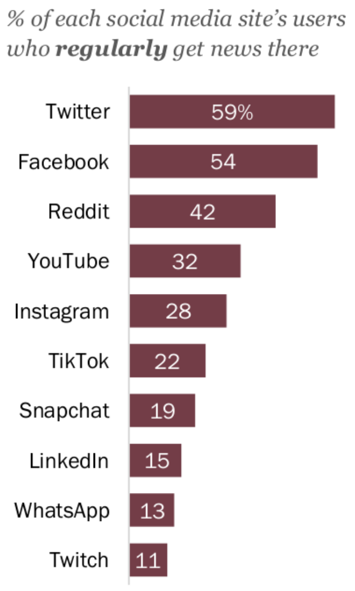


Figure 1: Large portion of Twitter users regularly get news on the site [8].

1.1 related work

Authors in [9] work on similar datasets but the tweets related to hurricanes. They get the data from Twitter and aim to first classify and then analyze disaster-related tweets. In their binary classification phase, classification methods: LSTM, CNN, SVM, Logistic Regression, and Ridge are performed, and based on their evaluation Long Short-Term Memory (LSTM) gets a better fit to the sequential order of textual data. Looking through the top solutions on the Kaggle website where they used LSTM structure to solve this task, we found and ran a notebook [1], which gave us a 0.78 F1 score. This is a decent score for an NLP task. However, based on a new method published by google researchers in 2018 called "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". The BERT can learn better contextual text [6]. Therefore, we are going to use this transformer for our task and try to get better F1 performance than LSTM and provide a comparison between them.

1.2 contribution

In this project, we propose a sequential model structure for this binary classification task based on the analysis we did on the dataset. In our proposed model the BERT layer is used as a trainable embedding layer. In addition, based on the discussion in the related works section, we also provide a competitive base model that has an LSTM structure. Our proposed model outperforms the base model in terms of F1 score by a five percent difference.

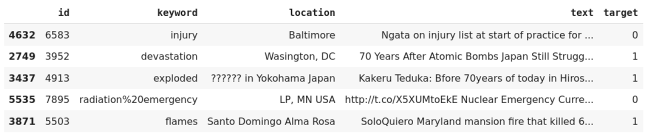
2 METHODS

2.1 Dataset

Each sample in the dataset has four columns:

- *id*: a unique identifier for each tweet
- *text*: the text of the tweet
- *location*: the location the tweet was sent from
- *keyword*: a particular keyword from the tweet
- *target* (label): in train data only, this denotes whether a tweet is about a real disaster (1) or not (0).

Fig. 2 shows five samples of data set. The type of dataset for columns *keyword*, *location*, and *text* columns are String, and the target (label) column is float64.



	id	keyword	location	text	target
4632	6583	injury	Baltimore	Ngata on injury list at start of practice for ...	0
2749	3952	devastation	Wasington, DC	70 Years After Atomic Bombs Japan Still Strugg...	1
3437	4913	exploded	?????? in Yokohama Japan	Kakeru Teduka: Bfore 70years of today in Hiros...	1
5535	7895	radiation%20emergency	LP, MN USA	http://t.co/X5XUMtoEkE Nuclear Emergency Curre...	0
3871	5503	flames	Santo Domingo Alma Rosa	SoloQuiero Maryland mansion fire that killed 6...	1

Figure 2: Data samples

Train data contains 7613 samples, and test data has 3263 unlabeled samples. The closest dataset to our dataset was used in [9]. Authors manually labeled 20,000 tweets keyword-based and geo-located corpus of tweets generated before, during, and after Hurricane Irma in its impact area. The dataset has the same features as our dataset, but this dataset is labeled based on only the Hurricane Irma disaster. However, our dataset is more generally related to various kinds of disasters.

2.1.1 *Dataset set analysis.*

Fig 3 depict the label distribution, about 57% of the data belongs to a non-disaster-tweet class (0) and the rest belong to a disaster-tweet class (1). Therefore, we do not expect class imbalances. A simple inspection showed there are no null values in labels.

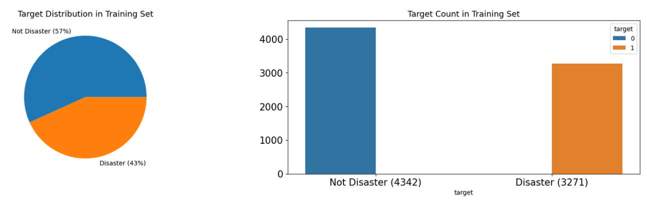


Figure 3: Target distribution in training set

We analyzed each input data features separately. The first feature of data is location. this column has lots of Null values and we replaced them with the value unknown. Fig. 4 shows the top 10 most frequently locations where the tweet was sent from. As it is clear, lots of data samples have *unknown* locations. Then using this feature does not help the model to get better results and it causes the model to mislead.

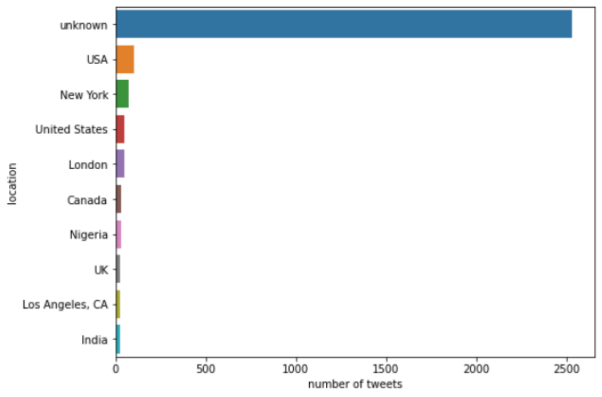


Figure 4: Location distribution in training set

The second feature will be the *keyword* feature. In contrast to what we had in the location column; the *keyword* column has some clues. The Fig. 5 shows the relation between *keywords* and *labels*. For example, the words: "debris", "derailment", and "wreckage" have been seen in only tweets that relate to a real-disaster class (1). Also, a word like "aftershock" is just used in tweets that point to a non-disaster class (0). In Fig. 5, *keywords* are sorted by average over labels for each *keyword*, which means that the leftmost word on the x-axis has an average of one (all *labels* corresponding to that *keyword* are equal to one and relate to a disaster). The *keyword* column has 61 null values over all train samples, which we can simply remove those samples from training data.

Finally, we did explore the *text* feature. The most challenging part of this task is cleaning the text column, where we deal with lots of stop words, numbers, dates, punctuation, accent characters, tags, web-links, etc. The preprocessing text has been done not only by removing all tags, web-links, punctuation, stop words, accents, and unknown ASCII characters, but also by converting abbreviated words. Word cloud for each class has been shown in Fig. 6.

Based on 6, somehow, we can get the sense that which one is related to which classes, and that is why other base model solutions that use simple embedding layers and deep neural networks can get a satisfactory performance. However, in this task, we are aiming one step further to use the BERT transformer that can capture the context relation between words in *text* input to get promising results.

2.2 Models

In this section, first we delve into our proposed solution and then we explain our base model.

2.2.1 *Proposed Model.*

We use the BERT transformer and update its parameter through the training process. There are couples of choices for the BERT transformer that we choose to work with the simple one that corresponds to uncased English words and its

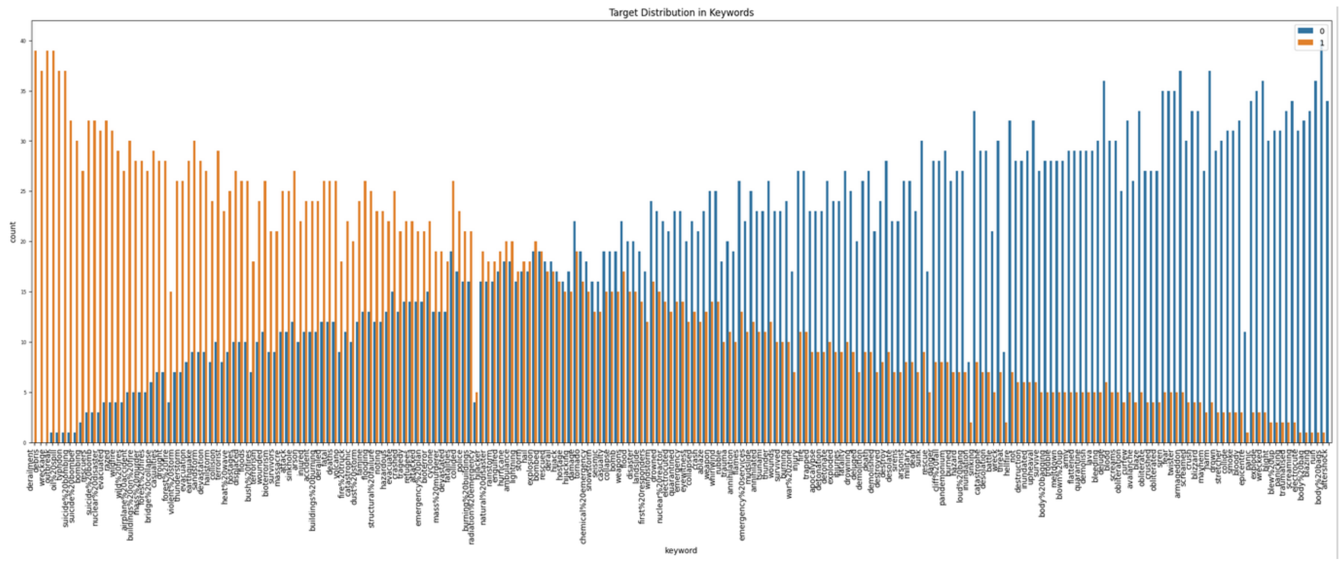


Figure 5: Target Distribution in Keywords

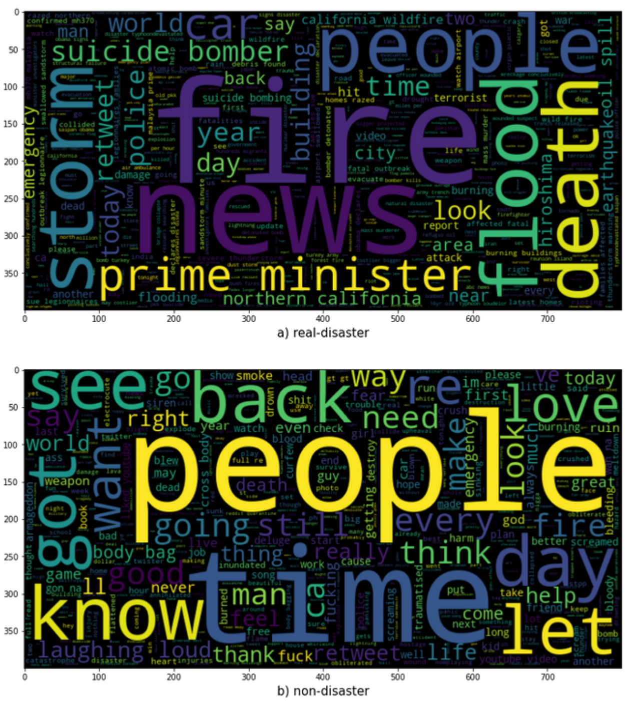


Figure 6: Word cloud after preprocessing *text* feature.

generated classification token ([‘CLS’]) has 768 entries for each data sample. The classification token generated from BERT is fed to a dense layer with 1024 units, and then the last layer has one unit for classification. Fig. 7 illustrates the structure of proposed model. Given the analysis in Dataset section, the features that are going to be used are the *text* (text input), and *keyword* column (categorical input) of each data sample. In [7], authors executed some

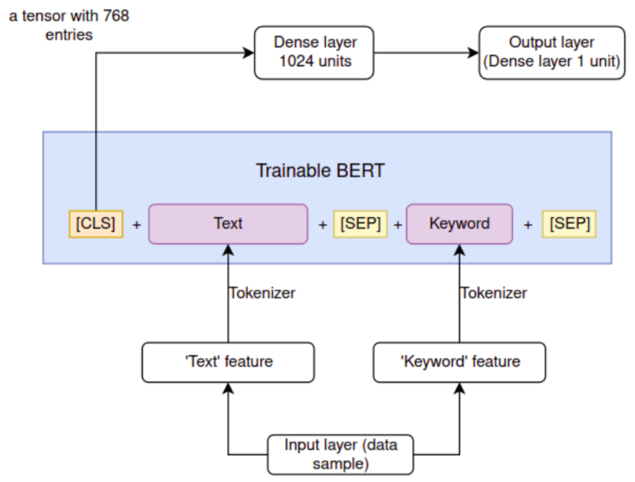


Figure 7: Proposed model structure.

experiments on BERT transformer with the dataset that includes various categorical and numerical features along with *text* feature. The authors experimentally proved that categorical features can be added to *text* feature as a new sentence to get a better performance. Therefore, as you can see in Fig. 7, we added our *keyword* feature to the *text* as second sentence (in BERT model [SEP] token is used to separate the sentences).

2.2.2 *Base Model.* The sequential model that includes the Long short-term memory (LSTM) layers is chosen as a Baseline. It goes without saying that this is a well-known baseline since the LSTM is very promising in learning sequential data such as text data. The structure of the baseline model is represented in Fig. 8.

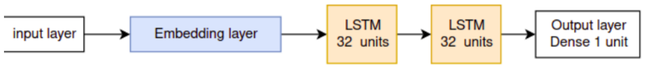


Figure 8: Base model structure.

2.3 Metrics

The evaluation method for this binary classification task is the micro-averaged F1 score between predicted and expected answers. F1 is calculated as follows:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}, \tag{1}$$

where

$$precision = \frac{TP}{TP + FP}, \tag{2}$$

$$recall = \frac{TP}{TP + FN}. \tag{3}$$

TP, FP, and FN stands for true positive, false positive, and false negative; respectively.

Given the nature of the task is disaster classification, we are more interested in having a perfect Recall which means that out of all positive records (real disaster) how many records are predicted correctly because missing a real disaster is more unfortunate than predicting a fake disaster. The advantage of choosing the Recall as an alternative metric is that we are less likely to miss a real disaster. However, the disadvantage of this metric over F1 score is that we will have worse precision.

3 RESULTS

3.1 Training setting

The implementation has been done by Tensorflow, and Keras libraries [5]. We used Colab-pro's GPU for the training process. The train and test datasets have 7613 and 3263 data samples, respectively; 20% of train data randomly selected to be the validation dataset. The training process has two epochs with the batch size equals to 32. The activation function for dense layer and output layer is set to *tanh* and *sigmoid* respectively. The *binary crossentropy* is chosen as the loss function. The parameters correspond to BERT transformer is set to be trainable during training process.

3.2 Hyper-parameter optimization

As specified by the original paper of the BERT [6], the optimizer used is Adam, with a leaning rate between 2e-5 and 5e-5. Since the BERT layer plays a key role in our model, the optimization of learning rate will be selecting the learning rate between 2e-4, 2e-5, 2e-6. In Tab. 1, the results for different learning is shown. The 2e-5 gives the best F1-score performance and we proceed the training process with this value.

As a second hyper-parameter in different experiments we try to find best regularization schemes for our problem, L1, L2, and L1&L2 are used for the hidden dense layer. L2-norm has slightly better performance as you can see in Tab. 2

Table 1: Adam's learning rate optimization experiment

Learning rate	F1-score
2e-4	0.57
2e-5	**0.83**
2e-6	0.80

Table 2: Regularization scheme optimization experiment

Regularization scheme	F1-score
L1	0.8176
L2(0.001)	0.8235
L2(0.01)	**0.8302**
L2(0.1)	0.8219
L1&L2(0.01)	0.8234

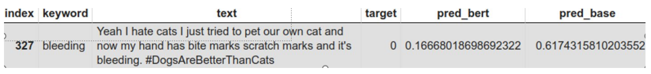
Table 3: Regularization scheme optimization experiment

Model	F1-score
Base model (LSTM)	0.78
Proposed model (BERT)	**0.83**

3.3 Final results

In the Kaggle competition correspond to this task, the micro average F1-score was considered as metric. F1-score comparison between the trained proposed model and base model for test dataset is shown In Tab. 3. Proposed model beats the base model on the same train, test, validation splitting dataset.

Finally, we provide a very good example to illustrate the strength of the proposed model. In Fig. 9 you can see how the proposed model can understand the context of the tweet where the base model failed to predict correctly and probably was deceived by the negative words inside the text (e.g. hate, bleeding, scratch, bite).



index	keyword	text	target	pred_bert	pred_base
327	bleeding	Yeah I hate cats I just tried to pet our own cat and now my hand has bite marks scratch marks and it's bleeding. #DogsAreBetterThanCats	0	0.16668018698692322	0.6174315810203552

Figure 9: Prediction example.

4 DISCUSSION

One big limitation with the dataset is that in some tweets there is not much information to tell. For example, e.g. "CHECK THIS PICTURE OUT!!!"; this tweet can be interpreted as a disaster or not a real disaster because it refers to a picture that is not provided in the dataset. From our standpoint of view, the dataset can significantly be improved by at least providing features like pictures, number of retweets, and number of likes or dislikes.

There is also an ethical issue with this dataset. Since the dataset contains text that may be used slang words, there is no clear solution to find out how are they distributed in the dataset. For example, if we assume in this specific dataset that slang words are most used for real disaster tweets, then the trained model will be slang biased and will predict the tweets that include the slang words as a real disaster tweet.

5 CONCLUSIONS

The disaster tweet classification was the task that we engaged in with this project. During dataset analysis, we illustrated that *keyword* feature has some correlation with the target and can be used along the *text* feature. Then, we came up with the structure that uses both features in the BERT transformer. For the sake of comparison, we also provide a base model with an LSTM structure based on what we discussed in the related work section. The proposed model structure outperforms the competitive base model by a about five percent improvement in terms of F1 score.

REFERENCES

[1] [n.d.]. LSTM Disaster Tweet Classification. https://www.kaggle.com/code/subrahmanyajoshi/cnn-lstm-disaster-tweet-classification.

[2] [n.d.]. Natural Language Processing with Disaster Tweets. https://www.kaggle.com/competitions/nlp-getting-started.

[3] [n.d.]. tweet: On plus side LOOK AT THE SKY LAST NIGHT IT WAS ABLAZE. https://twitter.com/AnyOtherAnnaK/status/629195955506708480.

[4] [n.d.]. Twitter Official Blog. March 14, 2011. https://blog.twitter.com/official/en_us/a/2011/numbers.html.

[5] Francois Chollet et al. 2015. *Keras*. https://github.com/fchollet/keras

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Ken Gu and Akshay Budhkar. 2021. A package for learning on tabular and text data with transformers. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*. 69–73.

[8] Elisa Shearer and Amy Mitchell. 2021. News use across social media platforms in 2020. (2021).

[9] Muhammed Ali Sit, Caglar Koylu, and Ibrahim Demir. 2019. Identifying disaster-related tweets and their semantic, spatial and temporal context using deep learning, natural language processing and spatial analysis: a case study of Hurricane Irma. *International Journal of Digital Earth* (2019).