Natural Language Processing with Disaster Tweets using Bert

Yulin Xue
y79xue@uwaterloo.ca
20949719
University of Waterloo

Abstract

In today's society, social media has become an important communication tool. People are tweeting in real time what they observed, including natural disasters. As a result, building a model that can monitors tweets related to natural disasters would be very helpful. However, it is not easy, a word may have a different meaning in different contexts. A word used to describe natural disasters can also be used in everyday life. Therefore, Bert, which can analyze the meaning of sentences, becomes very important in the model

1. Introduction

Bert

In Computer Vision, we can use a very large data set such as ImageNet to train a Convolutional Neural Network model, which can be used for different task from computer vision. Nevertheless, there is no model in Natural Language Processing that can perform such a task until Bert.

Bidirectional Encoder Representations from Transformers (Bert) is a pre-trained model that can be used to analyze the meaning of sentences. Bert was developed by by Jacob Devlin and his team from Google in 2018. (Devlin et al., 2018)

Traditional model such as Recurrent Neural Network only process sentence from left to right just like human are used to. However, the modern bidirectional model such as Bert provide a new view that right-to-left reading is just as important to machines as left-to-right reading. Based on that, during the pre-training stage, Bert masks some tokens in the sentence, and then let Bert to predicts these masked tokens.

Bert has greatly promoted the progress of natural language processing and has achieved very good results in different downstream tasks.

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Jul 24, 2021	{ANNA} (single model) *LG AI Research*	**90.622**	**95.719**
2 Apr 10, 2020	LUKE (single model) *Studio Ousia & NAIST & RIKEN AIP* https://arxiv.org/abs/2010.01057	90.202	95.379
3 May 21, 2019	XLNet (single model) *Google Brain & CMU*	89.898	95.080
4 Dec 11, 2019	XLNET-123++ (single model) *MST/EOI* http://tia.today	89.856	94.903
4 Aug 11, 2019	XLNET-123 (single model) *MST/EOI*	89.646	94.930
5 Jul 21, 2019	SpanBERT (single model) *FAIR & UW*	88.839	94.635
6 Jul 03, 2019	BERT+WWM+MT (single model) *Xiaoi Research*	88.650	94.393
7 Jul 21, 2019	Tuned BERT-1seq Large Cased (single model) *FAIR & UW*	87.465	93.294
8 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	87.433	93.160

Table 1: SQuAD1.1 Leaderboard result, from https://rajpurkar.github.io/SQuAD-explorer/

Disaster Tweets

Disaster Tweets is a dataset from Kaggle, which collect 7613 training data containing ID, keyword, location and text. Keyword is some particular word that has a potential connection with natural disasters (e.g. ablaze and aftershock) appearing in tweets, it could be blank. Location is the place where the tweet is posted, could be country, city, street and blank. Text is the message in tweet. It worth noting that text contains a lot of information in addition to English character such as hyperlink(e.g. https://....). It is necessary to remove these tokens since it might confused Bert.

The goal of this paper is to build a model that can use these information from tweets to make predictions about whether a tweet is a related to natural disaster. Disaster Tweets is also a Kaggle Competition, the test set can be used to verify the prediction result at the end.

2. Related Work

Word2Vec

Word2Vec is a method for natural language processing in token-level which is published by Tomas Mikolov in 2013. It is the first time that we could have math explanation on English word. Specifically, we can add or subtract words by using Word2Vec. For example, if x = ”King” - ”Man” + ”Woman”, then the word with the highest similarity to x in the thesaurus will be ”Queen” (Mikolov et al., 2013). However, Word2Vec also has some limitations, it cannot understand the meaning of the word based on the context. For example, Word2Vec cannot distinguished the difference of ”left” in the following context ”I left school” and ”My left hand”, because Word2Vec is on the token-level. On the other hand, Bert can distinguished the difference meaning of a word based on the context, because Bert is on sentence-level.

GPT

There are many application developed based on GPT in recent years such as chatbot and GitHub Copilot. GPT was developed by OpenAI team at June 2018, a lot of interesting perspective were made about natural language processing, part of it gave Bert's team some inspiration. Data has always been a challenge for natural language processing because there is so little labeled data. Natural language processing is not like image recognition, because the variation of language is much larger and unpredictable than that of pictures, and there is no efficient way to label a large amount of text accurately. This is also the reason why the progress of natural language processing has been very slow before. GPT proposes a different approach. They pre-trained on unlabeled text and fine-tuned the model on labeled text, which is also called semi-supervised. The final results are surprisingly good, before publishing the paper, they compared with other existing models on 12 different tasks, and the GPT model achieved better results than other models on 9 tasks (Radford et al., 2018) After that, they developed GPT2 and GPT3 in 2019 and 2020 respectively. They mainly greatly increased the size and parameters of the model, as a result, GPT3 also achieved a very good results.

One of the difference between GPT and Bert is that GPT used the decoder from transformers while Bert used the encoder. The decoder has the advantage of being able to translate and summarize text, while encoder is better suited for text classification. On the same scale, the accuracy of Bert is better than GPT. Bert-base has a very similar scale with GPT-1, but higher accuracy. This is also the reason why Bert is more popular in the industry, less hardware and memory requirements but better performance.

3. Data Exploration

The training dataset contains 4342 tweets not related to disaster and 3271 tweets related to disaster(0.57:0.43) which can be considered as a balanced dataset

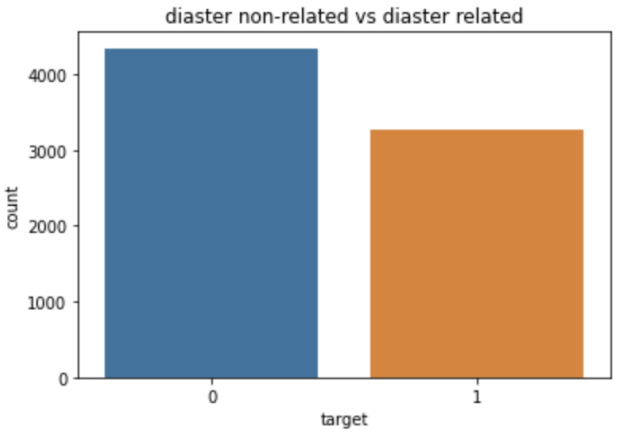

Figure 1

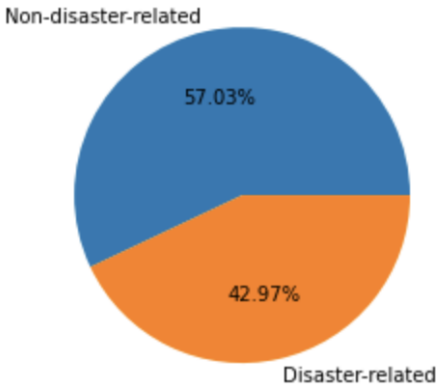

Figure 2

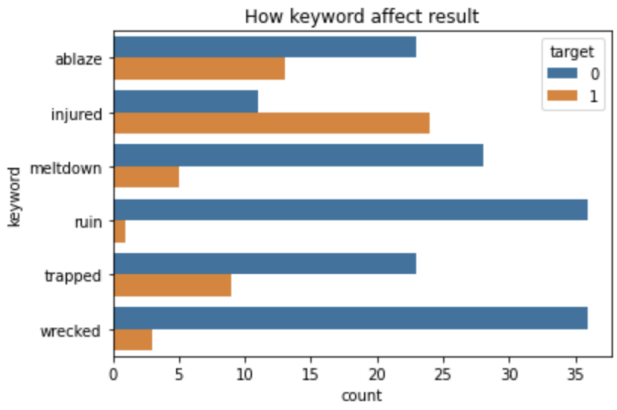

Figure 3: There are more than 6 keywords on the dataset, 6 keywords are randomly selected to form organized graph.

It is also worth noting that both training and test dataset contains feature ”keyword” that potentially could be used for prediction. As can be seen from Figure 3 below, natural disasters are more likely to be unrelated to tweets that use the word ”ruin”, while the word ”injured” is more likely to appear in a tweet related to natural disasters.

If we use a Convolutional Neural Network or Recurrent Neural Network model, we can directly use ”keyword” as the input of an meta-feature. However, it is not clear that how should we treat ”keyword” in Bert, the input of Bert is one sentence or a pair of sentences. It make no sense if we append the keyword to the beginning or the end of a sentences, since it could potentially change the meaning of the sentence.

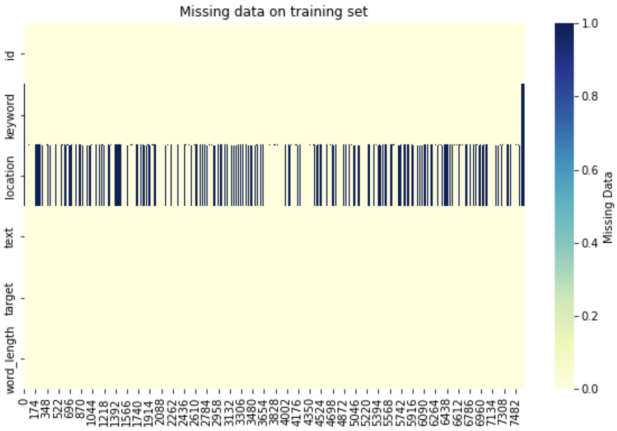

Figure 4a

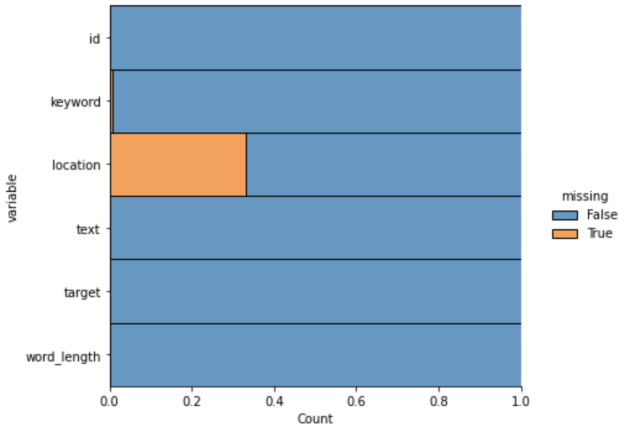

Figure 4b: There are missing value on location and keyword, only a very small amount of missing data on keyword(below 1%). The missing data on keyword are evenly spread among the training dataset(33%)

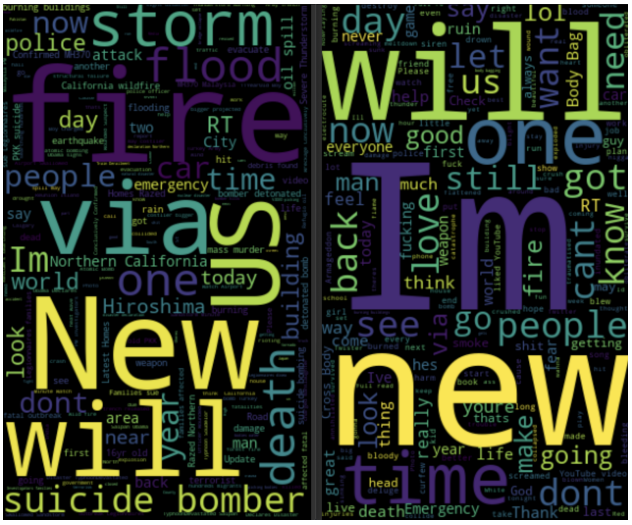

Figure 5: the frequency of token in disaster-related tweet vs disaster-unrelated tweet after data cleaning

Light data cleaning is performed before further processed. The hyperlink(e.g. https://) and special character is removed. However, there is still noise in text data. For example, the ’Im’ in the Figure 4 is what we get after removing ’ from ’I’m’

As can be seen from Figure 5, certain words appear more frequently in disaster-related tweets such as ”fire” and ”storm”. However, there are some high-frequency words that appear in both disaster-related and non-disaster-related tweets such as ”will”. Therefore, a token-level model like Word2Vec cannot be applied to this data set. Instead, we should apply a sentence-level model like Bert.

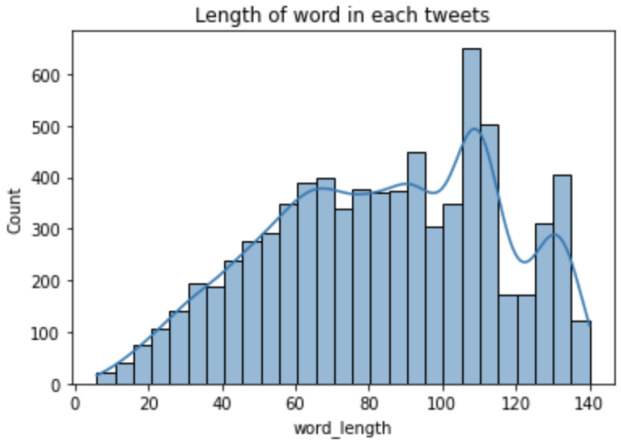

Figure 6: Number of word in a tweet (training dataset)

Verifying the length of input to Bert is always necessary, the maximum input sequence length is 512. This is because the memory requirements of the transformer model grow exponentially as input sequence grow. As can be seen, the maximum sequence length is 140, which is acceptable for Bert.

4. Model

Basically, Bert is the stack of transformer's encoder. Bert divides the input sequence text into three layers during prepossessing phase: Token Embeddings(CLS, input text and SEP), Sentence Embedding(text come from input sequence 1 or input sequence 2), Transformer Positional Embedding(Record the index position of each English characters). At the same time, during the preprocessing stage, Bert decides which words will be masked to train.

In the model of this paper, the embedding of the bert_base_uncased will be used directly, and then performed pre-trained and fine-tuned on top of this. Although Bert large has a better performance on a variety of task, it also required more input data. The dataset of disaster tweets is not large compared to the dataset used in Bert, so this means that using bert_large_uncased is not guaranteed to improve accuracy. At the end, we are also going to applied bert_cased model, to see if uppercase and lowercase characters in tweets have any effect on the results.

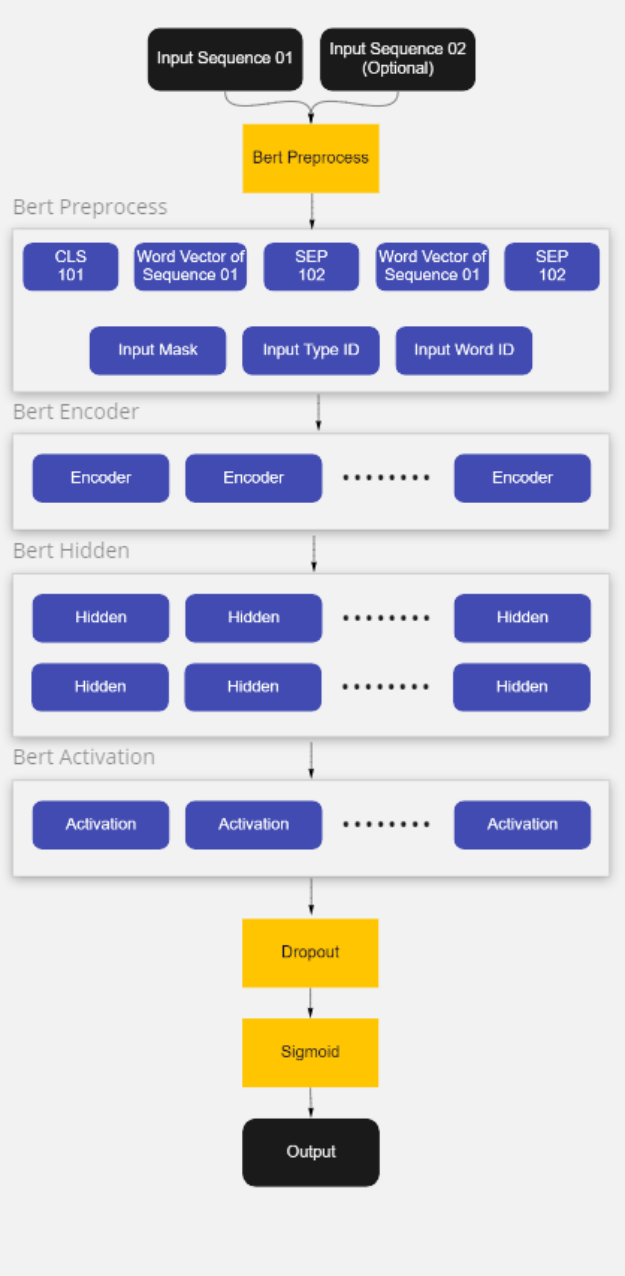

Figure 7: The model used in Disaster Tweet, since we used the bert-based, there are 12 encoders, 768 hidden nodes, 12 activations.

5. Evaluation

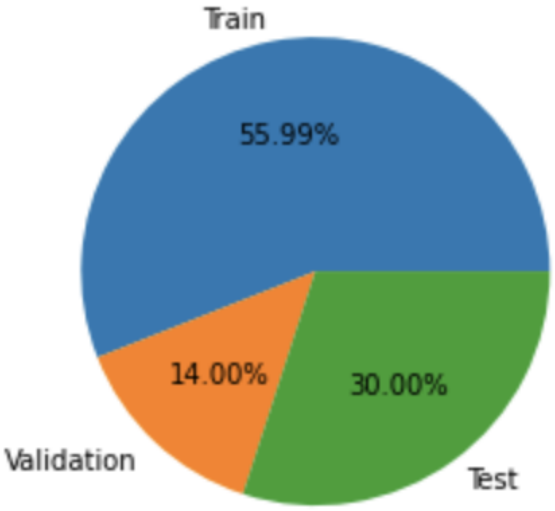

Figure 7: The public data is divided into training dataset and validation dataset according to the ratio of 80:20. Because this is a Kaggle competition, the correct results on the test data are hidden.

Bert			
Batch size	16	32	
Learning rate(Adam)	5e-5	3e-5	2e-5
Number of epochs	2	3	4

Table 2: The recommend parameter of Bert(Devlin et al., 2018)

Epoch	1	4	20
Training Accuracy	0.8064	0.9517	0.9837
Validation Accuracy	0.8352	0.7919	0.8142
Test Accuracy	0.8173	0.8029	0.7987

Table 3: Learning Rate(Adam) = 3e-5

Learning Rate(Adam)	5e-5	3e-5	2e-5
Training Accuracy	0.8057	0.8064	0.8020
Validation Accuracy	0.8280	0.8352	0.8365
Test Accuracy	0.8210	0.8360	0.8278

Table 4: Epoch = 1

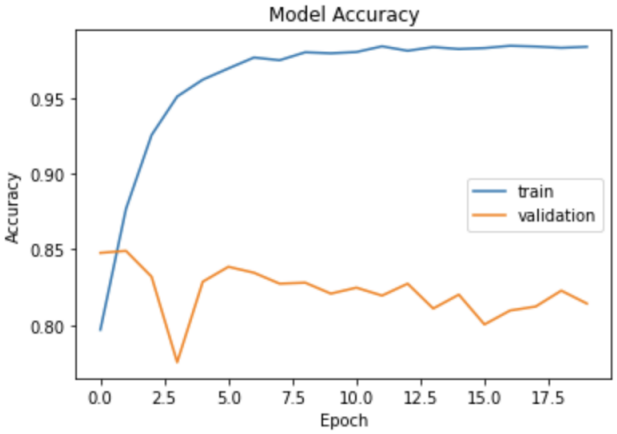

Figure 8

As can be seen from Figure 8 and Table 3, Bert converge very fast, it can achieve the highest validation accuracy during the first iteration. The validation and test accuracy are higher than training accuracy, because on the training phase, there is a 0.1 dropout rate.

Bert model

Bert	Based	Large	Cased
Training Accuracy	0.8018	0.7780	0.7931
Validation Accuracy	0.8168	0.8060	0.8106

Table 5

As can be seen from Table 5, the validation accuracy from 3 different Bert model are very closed. The validation accuracy of bert_large is slightly lower than bert_based, the reason for this might be that the amount of training data is insufficient. At the same time, from the results, it seems that capitalization has little correlation with whether Twitter is related to natural disasters. Based on this result, I will choose bert_based, as bert_based is the most efficient and more general than any of the others.

Pre-training

Even though Bert has done a lot of pre-training on Wikipedia and BooksCorpus, the performance of Bert can still be enhanced by pre-training it on a particular domain. As a result, we fetched a very similar dataset from Kaggle (VIKTOR S, 2021) which also contains information about tweets that can be used for natural disaster prediction.

Learning Rate(Adam)	Base	Pre-training
Training Accuracy	0.8057	0.8686
Validation Accuracy	0.8280	0.8621
Test Accuracy	0.8210	0.8241

Table 6

As can be seen from Table 6, the validation accuracy is 0.8621, which is significantly higher than any previous validation accuracy. However, test accuracy has not improved. It is possible that some of the data in the pre-training data overlaps with the data in the validation data set or maybe it's just normal fluctuations in the model

Fine-tuning

Based on a large number of experiments, it was found that fine-tuning is more important than pre-training for Bert's model, and it is easier to produce more accurate results. Since the information caught by each layer is different, the contribution of each layer to the result prediction is also different. Also, Bert can easily lead to overfitting. As shown in Figure 8, Disaster Tweet dataset has also been affected by overfitting. Differentiating the learning rate for different layers is a proven technique that may boost the accuracy for Bert (Chi et al., 2019)

The learning rate are updated as followed:

$$\eta^{k-1} = \xi * \eta^{k} \tag{1}$$

where η^{L} stands for the learning rate at Lth layer, ξ is the decay factor from 0 to 1. When $\xi = 1$, all the layer share the same learning rate. When $0 < \xi < 1$, the higher layer learns faster than the lower layer.

Fine-tuning	Adam	SGD
Training Accuracy	0.8108	0.4548
Validation Accuracy	0.8418	0.4826
Test Accuracy	0.8409	0.4683

Table 7: initial learning rate = 3e-5, decay factor = 0.9, Epoch = 1

After we applied this technique to Disaster Tweets dataset, it improve the accuracy on test dataset(from 0.8360 to 0.8409) which is also the best accuracy among all the configurations. It is also proved in the model of this paper that this is indeed a simple and effective method. The reason why SGD performs very poorly in the model here may be that the number of Epochs here is too small, resulting in the model not yet converging.

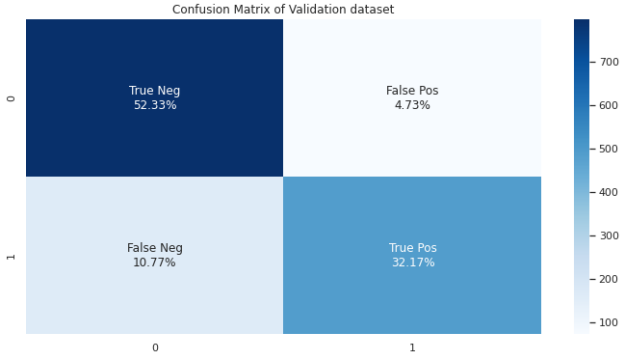

Figure 9: Threshold = 0.5

The model is more likely to predict a negative outcome. It is partly because there are more negatives observations than positives observations in the training dataset and it is partly due to the fact that there are more negatives than positives in the real results of the validation dataset

Best score

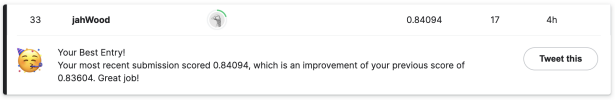

Figure 10: The model was ranked 33th out of a total of 812 teams participating in the Twitter Disaster contest in these two months.The test accuracy of 0.8409

The best score from leaderboard is 0.85136 without considering the score of 1. Since the test set of Disaster Tweet is public, some people get test data for training or even hard-code the prediction labels to achieve perfect predictions. Kaggle officials are also aware of this, but Disaster Tweet does not award ranking points or tiers, so they record are not removed from the leaderboard(https://www.kaggle.com/competitions/nlp-getting-started/overview/faq). The algorithm mentioned in this paper are quiet close to the best result with the test accuracy of 0.8409.

6. Conclusion

In 2018, natural language processing made a huge leap forward with the development of Bert and GPT. They each have their own strengths and weaknesses, but since then, countless studies and application have been developed based on them. Among them, there are countless research and applications based on Bert. This also has an impact on the traditional natural language processing model such as Recurrent Neural Network and Convolutional Neural Network model. Because when using Bert's embedded model, you don't need to spend too much time pre-training and it is likely to achieve better results than the traditional model.

7. Reference

Kaggle. Natural Language Processing with Disaster Tweets. Retrieved April 01, 2022 from https://www.kaggle.com/c/nlp-getting-started.

VIKTOR S. (2021, April). Disaster Tweets. Retrieved April 03, 2022 from https://www.kaggle.com/datasets/vstepanenko/disaster-tweets

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. (2018). Improving language understanding by generative pre-training (2018).

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to Fine-Tune BERT for Text Classification? arXiv:1905.05583 (2019)