NLP with Disaster Tweets

Group members: Xiaodong Li, Qian Lang
Kaggle Link: [Natural Language Processing with Disaster Tweets]
Our code implementation: https://github.com/qianlang95/disaster-tweets-predict

1.1 Introduction:

The project aims to detect disaster-related tweets, with deliverables including training various text classification ML models such as Logistic Regression and Bidirectional Encoder Representations from Transformers (BERT) to identify disaster-related tweets and comparing their performance. Its purpose is to leverage AI for filtering crucial and time-sensitive information from the vast volume of online social media content, while also serving as a learning opportunity for training tailored text classification ML models and gaining experience on platforms like Kaggle for potential future competitions. The applications of the project span disaster response, emergency alerts, and disaster pattern research among others.

1.2 Previous Work and Contributions

Paper Research

Paper 1: VictimFinder: Harvesting rescue requests in disaster response from social media with BERT

- Summary
  - This paper studies ten ML models (3 based on milestone NLP algorithms, 4 BERT-based, and 3 customized BERT-based models) trained to identify rescue request tweets. Experiment results show that BERT with customized classification e.g. Convolutional Neural Network (CNN), outperforms the baseline models.
- Insights (what I liked about this paper)
  - Among 10 models, each model is trained using a different combination of **pretrained models** (e.g. Global Vectors for Word Representation (GloVe), Embeddings from Language Models (ELMo), BERT, RoBERTa, DistilBERT, ALBERT, and XLNet), **classifiers** (e.g. Linear, Nonlinear, Long short-term memory LSTM and CNN) and **training methods** (e.g. Feature-based, Fine-tuning). This is highly beneficial for finding the best-performing model for a specific task.
- Limitations
  - Lack of **generalizability** due to limited dataset which includes only tweets from a single event, Hurricane Harvey. Dataset including other hurricane events and other types of disasters should be used for model training as well.
  - The model only works for English-based tweets. Other languages are not supported.

Paper 2: [Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model]

- Summary
  - This paper proposes a regional CNN-LSTM model consisting of two parts: regional CNN and LSTM to predict the VA ratings of texts.
- Insights (what I liked about this paper):
  - By capturing both local (regional) information within sentences and long-distance dependency across sentences, this model combines CNN and LSTM to both capture local relationships and have some support for long term memory.The proposed method outperformed regression- and conventional NN-based methods presented in previous studies.
- Limitations
  - RNN processes a word one at a time sequentially. LSTM has a longer reference window than RNN. However, the window reference is still limited compared with the Transformer model.
  - The target language is only English and Chinese.

Contribution

The contribution of this project lies in its comprehensive examination of various machine learning models, coupled with diverse data preprocessing methods, for the task of disaster tweet detection. We explore Logistic Regression incorporating CountVectorizer and Term Frequency - Inverse Document Frequency(TF-IDF), BERT, BERT+CNN, and DistilBERT. Notably, separate training sessions were conducted for BERT, which employed distinct parameter configurations to facilitate thorough performance evaluation.

1.3 Methodology and Results

Explore the Dataset

This is the first 5 items of the train data:

	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1

The train dataset information in general:

```
RangeIndex: 7613 entries, 0 to 7612
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        7613 non-null   int64
 1   keyword   7552 non-null   object
 2   location  5080 non-null   object
 3   text      7613 non-null   object
```

```
 4   target    7613 non-null   int64
dtypes: int64(2), object(3)
memory usage: 297.5+ KB
```

We notice the shape of our dataset is (7613, 5). This is a relatively small dataset for fine-tuning a BERT model. Also, we decided to drop the keyword and location column due to too many empty values. In addition, the dataset is fairly balanced (4342 non-disaster tweets vs 3271 disaster tweets) and will be acceptable for our model.

After visually comparing the most frequent words in the disaster and non-disaster datasets, we observed that the disaster dataset contains more indicative words like fire, flood, and Hiroshima, which appear more frequently. These indicative words are crucial for our classification task. However, it's worth noting that the non-disaster dataset also includes these indicative words, which can be misleading. Therefore, we need to develop a model that can accurately determine whether a tweet genuinely relates to a disaster, such as 'Wholesale Markets ablaze,' or is misleading, like 'Crying out for more! Set me ablaze.'.

Explore the ML Models

Model Description

1. Baseline model: Logistic Regression
    a. Logistic Regression is a supervised machine learning algorithm widely used for binary classification tasks. We implemented it using the LogisticRegression module from the sklearn.linear_model library.
    b. Feature extraction techniques
        i. TF-IDF: a measure of importance of a word to a document in a corpus.
        ii. CountVectorizer: a class in scikit-learn that transforms a collection of text documents into a numerical matrix of word or token counts.
2. BERT related models
    1. *BERT*
        a. This fine-tuned model consists of a straightforward architecture, incorporating the preprocessing model (bert_en_uncased_preprocess), the chosen BERT model (small_bert/bert_en_uncased), a Dense layer and a Dropout layer.
        b. Fine-tuning Parameters to observe their effects
            1. Dropout == 0.1 vs 0.2
                - The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting.
            2. Learning Rate == 3e-5 vs 5e-5 vs 2e-5
                - The learning rate determines the step size at each iteration while moving toward a minimum of a loss function. In line with the BERT paper, the initial learning rate starts with a smaller value for fine-tuning (best of 5e-5, 3e-5, 2e-5).

2. *BERT+CNN*
    a. The text data undergoes processing via BERT(small_bert/bert_en_uncased), followed by the application of convolutional layers to capture local patterns. Dense layers are then employed for classification. The sequence_output (i.e. vector-space representations) from BERT, representing contextual embeddings for each token in the input sequence, serves as the input to the convolutional layers. This choice allows the convolutional layers to capture local patterns and relationships within the token embeddings.
3. *DistilBERT*
    a. The DistilBERT model is a distilled version of the BERT model. Through knowledge distillation during pre-training, the size of a BERT model was reduced by 40%, while still maintaining 97% of its language understanding capabilities and being 60% faster.

Model Statistics

ML Model	Parameters	Train Set			Validation Set		
		Loss	Accuracy	F1 Score	Loss	Accuracy	F1 Score
Logistic Regression	TF-IDF	N.A	0.88	0.85	N.A	0.81	0.76
	CountVectorizer	N.A	0.99	0.99	N.A	0.81	0.76
BERT	Dropout = 0.1; Learning RATE = 3e-5	0.38	0.84	0.84	0.40	0.83	0.80
	Dropout = 0.2; Learning RATE = 3e-5	0.40	0.83	0.83	0.41	0.83	0.79
	Dropout = 0.1; Learning RATE = 5e-5	0.36	0.84	0.85	0.40	0.84	0.81
	Dropout = 0.1; Learning RATE = 2e-5	0.28	0.88	0.89	0.44	0.84	0.80
BERT+CNN	Dropout = 0.1; Learning RATE = 3e-5	0.40	0.83	0.81	0.41	0.82	0.80
DistilBERT		0.38	0.84	0.86	0.38	0.85	0.81

Table 1: Statistics (Notes: Confusion Matrices are attached in the Appendix for reference)

Model Performance Analysis

We selected Logistic Regression as our baseline model. Additionally, we examined two feature extraction techniques. Interestingly, the performance of CountVectorizer was comparable to TF-IDF, which is typically considered a more optimized preprocessing method. This could be due to the importance of capturing disaster-related words, which CountVectorizer seems to handle effectively. We also noticed that the extremely high accuracy is caused by overfitting during the training process. However, the F1 score for the validation set shows relatively poorer performance compared to all other BERT-related models.

Among all the BERT related models, DistilBERT demonstrates a better performance (i.e. low loss coupled with high accuracy and F1 score for both train dataset and validation dataset). This result is unsurprising given DistilBERT's smaller and lightweight architecture, requiring less parameters for fine tuning. However, due to significant computational constraints, our project

exploration was restricted to a limited range of parameters for BERT and BERT+CNN models. As such, we may not train the models with the most optimal parameters, potentially hindering their performance compared to DistilBERT.

In our investigation of BERT-based classifier modes, we explored the model performance by fine tuning with different parameters. Specifically, we examined the impact of dropout rate, a technique aimed at preventing overfitting. However, the effect was negligible on the disaster tweets prediction dataset, with differences in metrics of approximately 0.01 only. One possible reason may be the dataset's small size and balanced nature, making it simple for model training and diminishing the need for dropout rate tuning. We also explored the impact of learning rate, which can lead to model overshooting with excessively large step sizes or slow convergence with overly cautious optimization. However, no single learning rate significantly outperforms others. While a low learning rate (2e-5) has better performance on the train sets, a high learning rate(5e-5) yields better results on the validation sets. It's worth noting that the BERT model was trained using the AdamW optimizer, which dynamically adjusts the learning rate during training based on the optimization process. This may potentially help to mitigate the impact of learning rate on model performance.

When comparing BERT-based classifier model and BERT+CNN classifier model, integrating CNN with a fine-tuned BERT model might improve the model performance by leveraging the strengths of both models. However, this anticipated improvement was not observed in the experimental results of this project. While CNN excels at capturing local patterns, disaster tweets are relatively simple and straightforward. As such, combining CNN with BERT may not yield a substantial performance improvement.

1.4 Conclusion

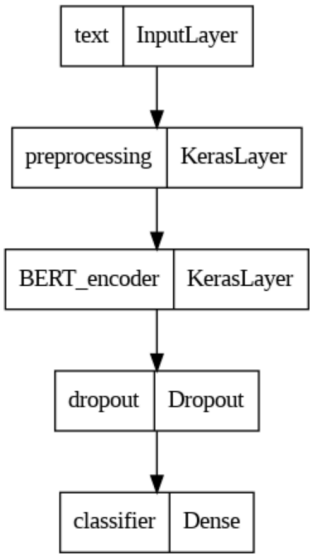
In this project, we explored various machine learning models (e.g. Logistic Regression, BERT, BERT+CNN, and DistilBERT), coupled with diverse data preprocessing methods (e.g. CountVectorizer and TF-IDF), for the task of disaster tweet detection. Among all the models, DistilBERT outperforms others across various metrics, given DistilBERT's smaller and lightweight architecture while retaining most of its language understanding capabilities.

For potential future work: Firstly, given that our dataset is relatively small for BERT, we are keen to collect more data to enhance our model's performance. Additionally, considering the computational demands of our work, we aim to leverage high-performance computing (HPC) and networking for model training.Moreover, although our current dataset is well-balanced and yields promising prediction results, we are interested in evaluating our model's performance on real-world data, which may be imbalanced and contain more noise. Lastly, we plan to incorporate a helpful feature, RAG (Retrieval-augmented generation), into our model. RAG enhances the accuracy and reliability of generative AI models by integrating facts from external sources. We believe that by proactively sourcing disaster-related and misleading information and feeding it into our BERT model, we can further refine and strengthen its capabilities.

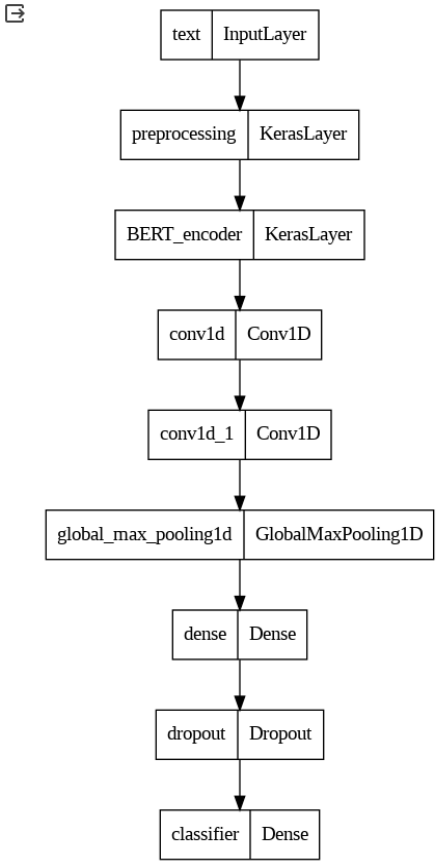
Appendix

Model Architecture

BERT

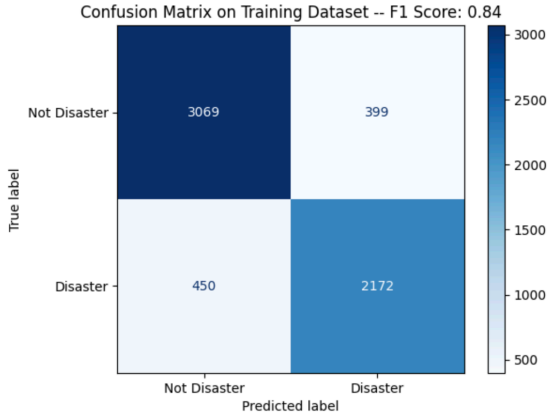


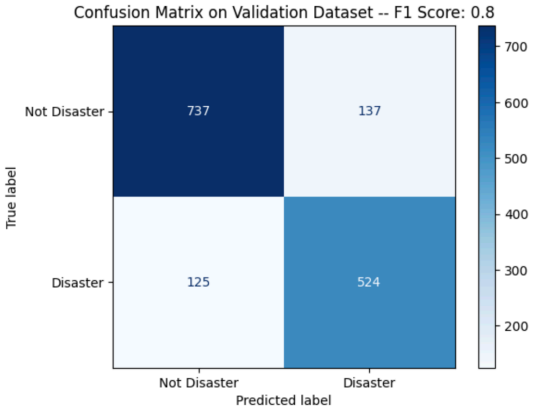
BERT+CNN



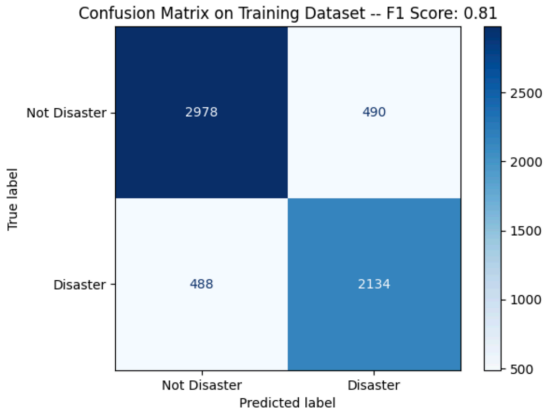
Confusion Matrix

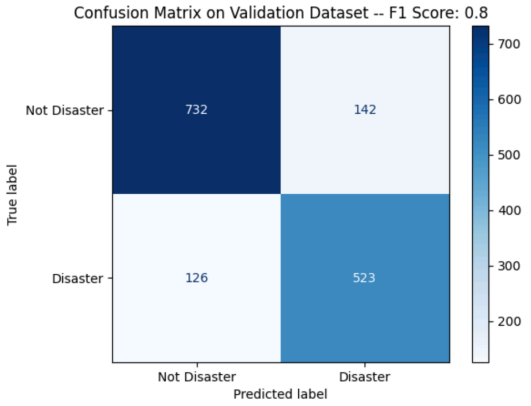
BERT (Dropout = 0.1; Learning RATE = 3e-5)



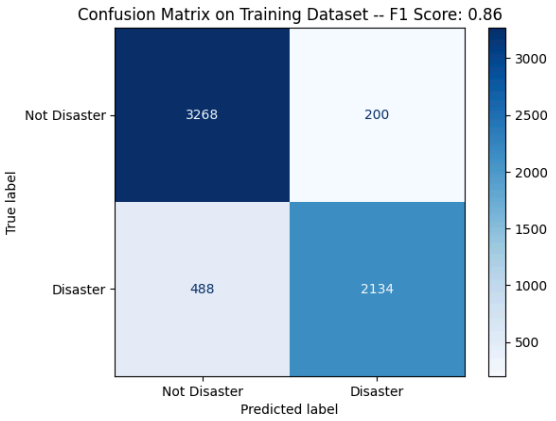


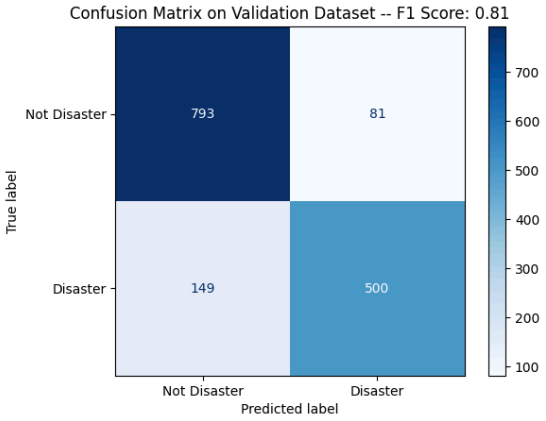
BERT+CNN





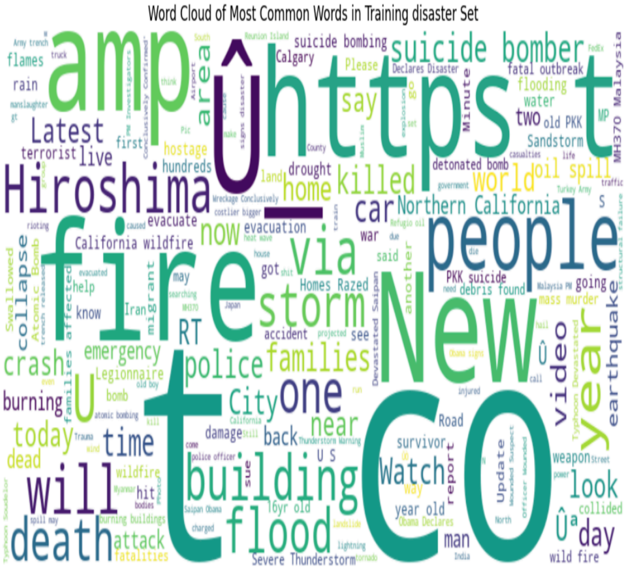
DistilBERT





Word Cloud

Word Cloud of Most Common Words in Training disaster Set



Word Cloud of Most Common Words in Training non-disaster Set