ML Final Project Report

Real or Not? NLP with Disaster Tweets

B06705024 郭宇軒

B07705029 林俊諺

B07901062 陳泊均

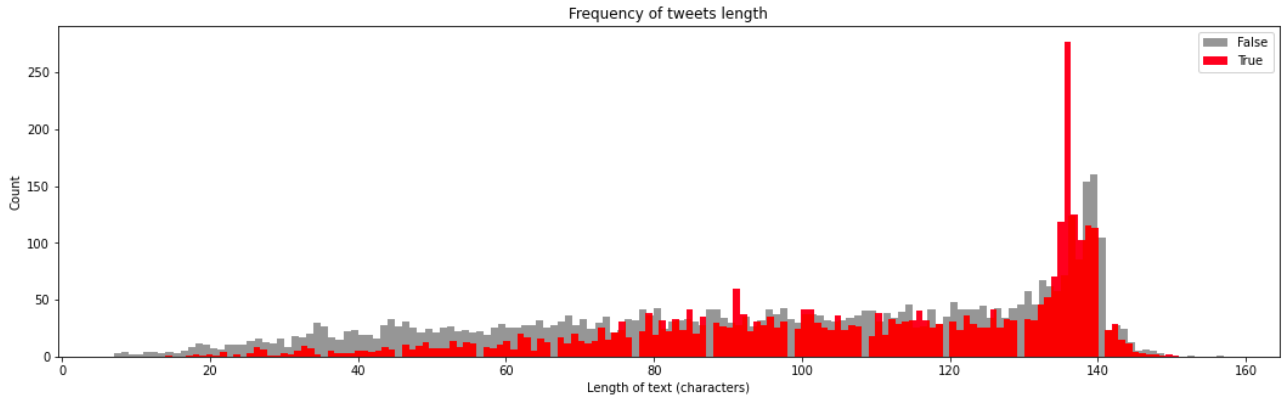
1. Introduction and Motivation

Twitter 推特是全世界最熱門的社交網站之一，全球活躍用戶目前大於 3 億人，而其特色是每篇發文 (tweet) 字數需少於 280 字，並且每篇 tweet 都可以附加 keyword 以及標註 location。

由於 Twitter 方便的特性，很多有關災難（Disaster）的發文會出現在 Twitter 上，而[Real or Not? NLP with Disaster Tweets] 這個 kaggle competition 便是需要我們根據一篇 tweet 的內文、keyword 以及標注的 location 來判定一篇 tweet 的內容是否在描述真的 disaster 。

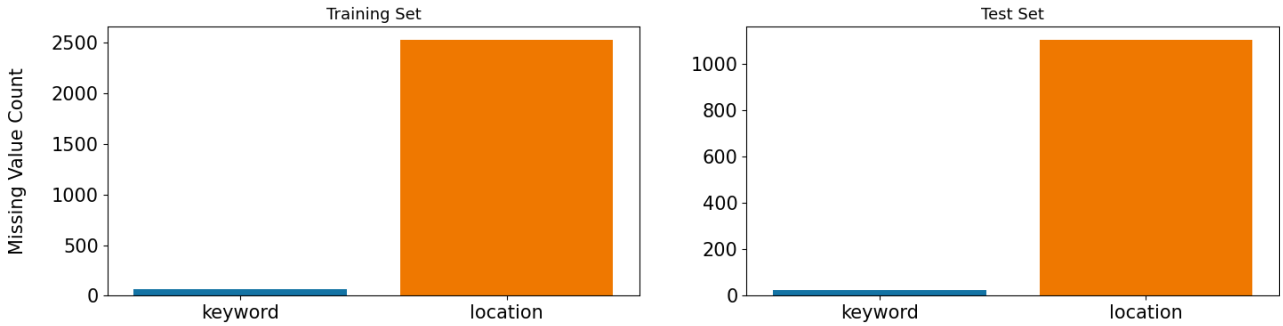
2. Data Preprocessing/Feature Engineering

我們在觀察資料集時，先觀察每則 tweets 的資料長度。如圖一，可以發現是 disaster 的 tweets 字數都集中在 80-140 個字之間， 不是 disaster 的 tweets 字數則分布的比較平均。



（圖一）每則 tweets 的 文長直方圖

確認資料完整度時，如圖二，我們發現 Training data 中總共有 7613 條 tweet，其中 0.8% 的資料缺少 keyword、33%的資料缺少 location； Testing data 中， 資料遺失的比例也和 training data 的比例差不多， 因此我們先判定兩者的資料來源相同，沒有太多的 bias。

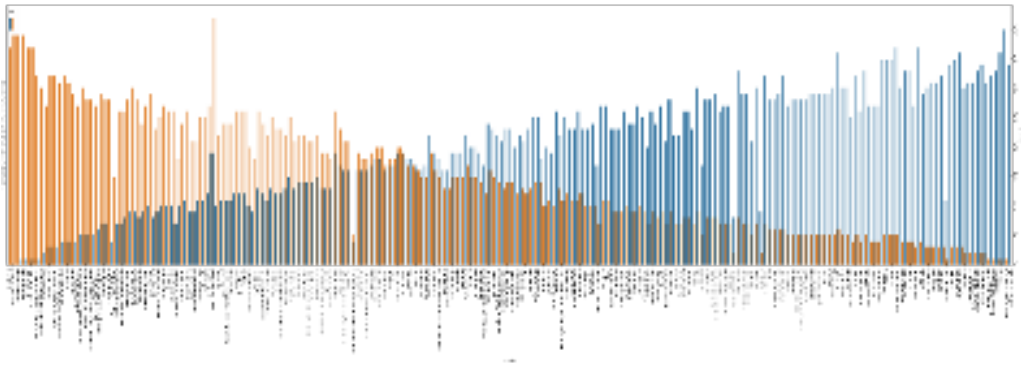


（圖二）Training 和 Testing Data 中的 missing value 比較

由於 Tweet 中的 Location 在 Twitter 中是可以讓發文者自由編輯的，因此 7613 筆資料 training data 中包含了 3342 筆 unique value，也包含了許多 dirty data 如 “Milky Way”, “銀河系” 等字眼。

假如使用 Location 作為 feature 應該要做一些 preprocessing (ex : Real life locations)或是考慮根本不使用，最終在 進行 Improved 的 training 時，我們也決定捨棄這項 feature，以此提升 model 的 準確率。

我們也發現 training data 中的 Keyword 總共有 212 種，都和 disaster 有相關 ，如 typhoon, outbreak, etc. 每個不同的 keyword 和災難的相關程度有高有低，如下圖中橫軸每個點都代表一個不同的 keyword，橘色直方代表多少篇 tweet 為真實 disaster；藍色直方則相反。因此我們可以看到圖表兩側中的 keyword 的出現代表高機率有真實災難或無災難，圖表中間的 keyword 則是模稜兩可。



（圖三）不同 Keywords 和災難真實發生相關性比較

3. Model Description

- Simple model

我們的 simple model 將每個字詞都用 Scikit-Learn 的 CountVectorizer 轉成詞向量， 並假設這些詞向量和我們的預設結果為線性相關，因此使用 Scikit-Learn 的 RidgeClassifier 線性模型，產生 kaggle public score : 0.78118 的結果，成功通過 Simple Baseline。

- Improved Model

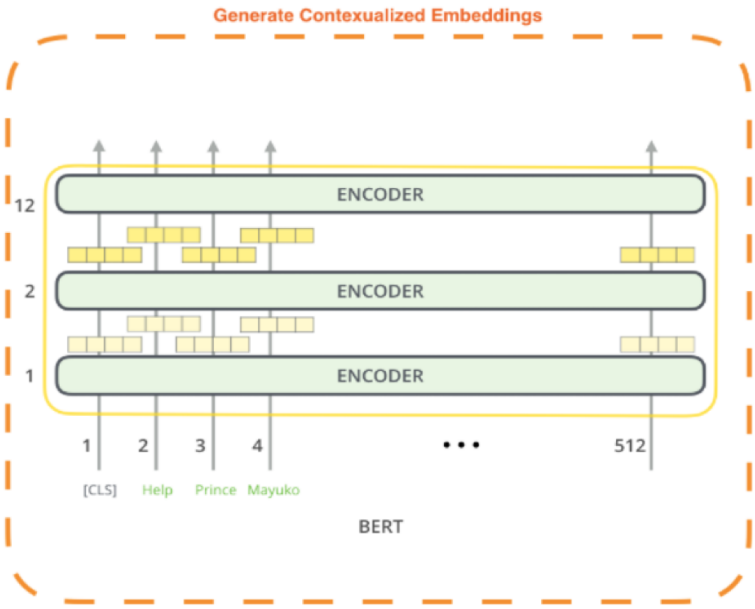
Simple Model 雖然可以為我們帶來不錯的預測結果，但依舊有許多提升空間。由於 BERT (Bidirectional Encoder Representations from Transformers) 是目前 NLP 問題最有效的方法之一，因此我們決定改用 pre-trained BERT 的相關資源來進行訓練，最終選定並專門進行 tweets 分析的 BERTweet Model.

而用來訓練 BERTweet 的資料集是由 850M 英文 tweets（16B ~ 80GB word tokens）所組成， 因此我們也改用 GloVe Pre-trained Corpus。最終，我們在 kaggle public score 得到 0.8432 的結果， 相較 simple model 獲得了一定程度的進展。

BERTweet Model 介紹：

BERTweet model 是第一個針對英文 tweets 進行訓練的公開 pretrained model, 是根據 RoBERTa 的結構再進行延伸和訓練的 Bert Based Model。

結構：

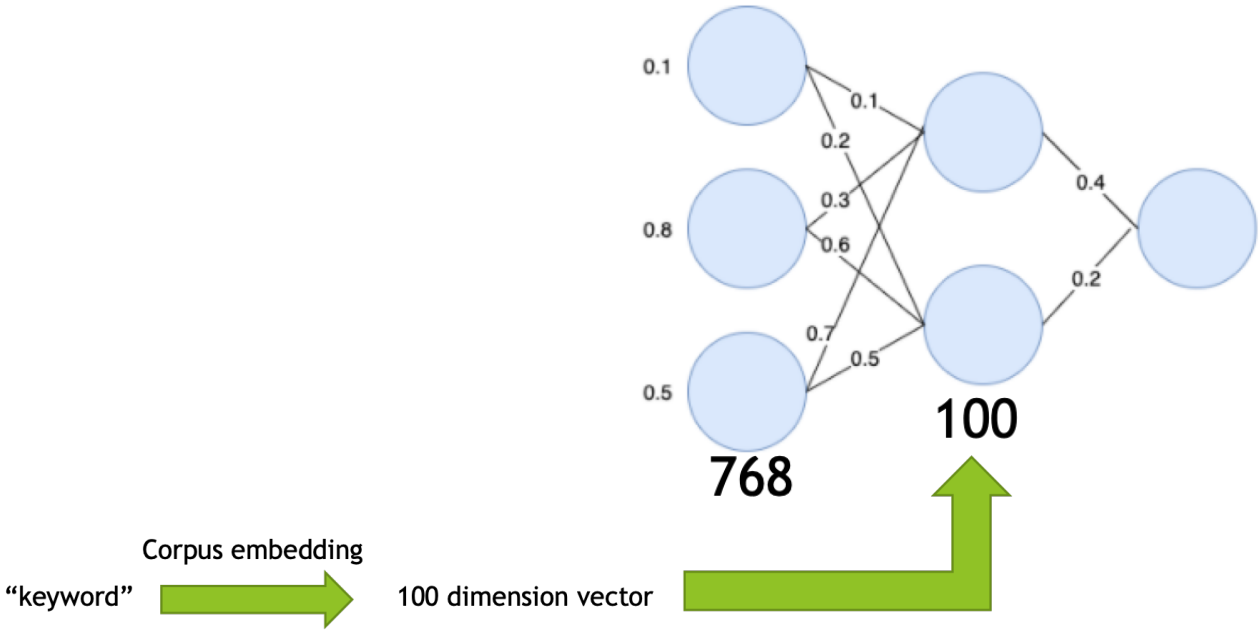


經過 BERTweet 後，我們集會得到 768 維的 output tensor。

我們的 Improved model 結構：

BERTweet	vinai/bertweet-base
Linear	Dropout(0.3)
	Linear(768 , 100 , bias = True)
	Sigmoid()
	nn.Linear(100 , 1 , bias = True)
	Linear(32,2)

在 Linear 的結構中，我們先將 BERTweet 所產出的 768 維向量 降為 100 維，接著將 100 維的向量加上該則 tweet 的 的 keyword 的 100 維 corpus embedding ， 如圖四。



（圖四）Improved Model 中的 Linear 結構概念。

完成了以上兩個 model 後，我們也比較了兩者之間的優缺點，如下。

Simple Model :

Pros:

- Simple，結構簡單，非常容易建立模型
- Training 幾乎不需要時間
- 就算 model 非常簡單預測結果也有一定水準

Cons:

- 也是 Simple，太過於簡單沒有什麼進步和調整的空間。
- Accuracy 還有一些提升空間（Only 0.78118 ）

Improved Model :

Pros:

- 利用 pertained Bert 準確率提升（0.8432）
- Deep Learning Model 可以學到的特徵較多

Cons:

- Data Preprocess 的空間受到限制
- Training 時間較長

4. Experiment and Discussion

- GloVe

GloVe 的全名叫 Global Vectors for Word Representation，它是一個基於全局詞頻統計（count-based & overall statistics）的 word representation 工具。我們利用 glove.6B 來準備為 keyword 進行 word embedding。

- Scheduler

我們利用 ReduceLROnPlateau 的方法來隨 epoch 調整 learning rate。只要 Validation Loss 降低幅度減少時，即減少 learning rate，幫助 model 收斂到最佳位置。

-Loss and Weight

原本我們先將Fully Connected layer通過Sigmoid function後，再使用BCE loss來進行training，但是這樣的結果並不太好，validation lost從第2個epoch就會開始下降。

因此後來我們把Fully connected layer的結果不經過Sigmoid function，直接使用BCEWithLogitsLoss來進行training。根據Pytorch 對BCEWithLogitsLoss的documentation," This loss combines a Sigmoid layer and the BCELoss in one single class. This version is more numerically stable than using a plain Sigmoid followed by a BCELoss as, by combining the operations into one layer, we take advantage of the log-sum-exp trick for numerical stability." 而我們也觀察到validation lost不會馬上下降，而是會在約6個epoch之後正常的收斂，訓練出來的accuracy也有一些提升，的確優於使用BCE loss。

而 BCEWithLogitsLoss 中的 pos_weight argument 可以對 positive and negative sample 進行加權，由於我們觀察到 label 的分布並不是 1:1，0 和 1 的比率大概是 1.3 左右，代表不是災難的比較多，因此 model 可能會傾向於保守的猜 0 這樣 loss 比較低，因此我們測試兩個不同的 pos_weight 數值，分別是 1.3 和 2，發現兩者都比不加 weight 的表現好，而後來選擇使用 pos_weight=2。

5. Conclusion

BERT 雖然是一個相當強大的 pretrained model，但是也因此很容易 overfitting，因 此在 training 過程必須十分注意，可以適當的使用 dropout 以及調低 learning rate 來減緩 overfitting 的趨勢 。

對於 BERTweet 而言，tweet 的原文都是有用的資訊，因此我們發現不做任何 preprocess 反 倒會有更好的表現，且 Keyword 對於 model 判斷一則 tweet 的災難相關性是十分重要的，將 keyword 放入 model 中，結果即有顯著的成長；在使用如 BERT 這般結構已十分複雜的 pretrained model 時，不管後面連接構複雜的 layers 或單純直接將其輸出都沒有很大的差別，推測可能是 BERT 的參數量已十分巨大，因此在其後調整模型的架構不會對整體的準確度有太大的提升。

6. Reference

ref : https://www.kaggle.com/anushakarthik1991/nlp-with-disaster-tweets-eda-cleaning-and-bert

ref : https://www.kaggle.com/philculliton/nlp-getting-started-tutorial/data?select=train.csv

ref : https://www.kaggle.com/anushakarthik1991/nlp-with-disaster-tweets-eda-cleaning-and-bert#7.-Model