Machine Learning for Natural Langage Processing

Allouche Dan
ENSAE Paris

Palaiseau, France
dan.allouche@ensae.fr

Dahan Nicolas
ENSAE Paris

Palaiseau, France
nicolas.dahan@ensae.fr

***Abstract*—Twitter has become an important communication channel in case of emergency. As a result, more and more organizations are becoming interested in programatically monitoring Twitter. Our objective is to classify whether a given tweet is about a real disaster or not.**

I. Introduction

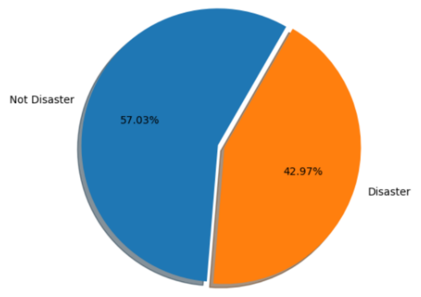
This work is done in the context of a Kaggle challenge [3]. Nevertheless our goal is not to get the best score in this challenge but to study and compare different Transfer Learning models, specific to NLP.

II. Data

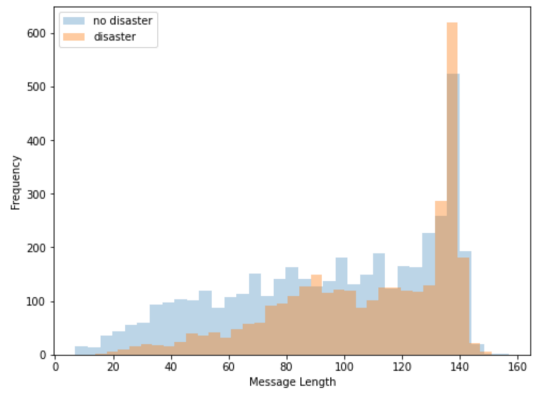
To achieve our goal, we use a data set created by the company figure-eight and originally shared on their 'Data For Everyone' [4]. The are 10873 tweets in the data set and it composed as follow:

```
          Features of the dataset
--------  ----------------------------------
id        a unique identifier for each tweet
text      the text of the tweet
location  location of the tweet
keyword   particular keyword of the text
target    1 : real disaster ; 0 : false
```

The aim is a binary classification task and the balance of the target is as follow :



The two categories are well represented and the balance is good.
More of that, the tweets' length according to each category has approximately the same frequency :



III. Models

We want to compare two different methods to perform Transfer Learning in NLP for classification tasks. The first method is the **features extraction**, which consist of using the a pre-trained version of DistilBERT without changing weights subsequently. We directly collect the embedding vectors of each token of the data set. Then we select the [CLS] token of each sentence (it's a special token created by the authors of BERT for classification tasks) and train different classifiers. We train a Logistic regression and a Random Forest. The second method is the **fine-tuning** of DistilBERT on the dataset.
We implement the Logistic regression and the Random Forest with Scikit-learn, and the fine-tuning with Pytorch.
We choose DistilBERT model (distil version of BERT [1], [2]), which is smaller and faster than BERT. We read on the literature that it is a good option for speed up calculations and achieve convergence with the features extraction approach.
For the training, we keep the 3000 first sentences (which keep a good balance) and divide in train set and validation set (the test set is not available because it's a challenge). The reason of this move is due to the first Transfer Learning approach which is very slow. With the Hugging Face library there is no possibility to collect just the [CLS] token. We need to "upload" all embedding vectors then select the [CLS] token (which is 768 dimension vector). More of that the grid search part for the two Machine Learning models is very slow. For this reason we do not keep more sentences and do not work with more tokens.

IV. Results

As it was imposed in the Kaggle challenge, we used F1-score to evaluate ours models. We also add more metrics to have a better interpretation of our results.

```
====  ==========  ========  ===========  ==========
..      accuracy    recall    precision    f1-score
====  ==========  ========  ===========  ==========
LR          0.78      0.75         0.78        0.76
RF          0.77      0.72         0.79        0.73
FT          0.8       0.76         0.84        0.77
====  ==========  ========  ===========  ==========
```

For each metric, the fine-tuning method perform the features-extraction approach. According to the literature this is not surprising.

V. Conclusion and future work

We saw, in our case, that the fine-tuning method represent the better option to do binary classification. More of that, the computation time was much faster with the fine-tuning approach. The method with machine learning algorithms was very slow and push us to restraint our work on a subset of our data set which is the one selected for a Kaggle challenge. So we should have used, in a logical spirit of competition - it's open to all challenge - the all data set. But in our case, we were interested in the models and not in the competition. For future work, it would be relevant to implement the Logistic regression with Pytorch instead of Scikit-learn. With the dimension of one embedding vector and calculations, the use of GPU is very appropriate. In fact this helped us a lot to realize the fine-tuning. If we manage this problem, for the feature extraction approach, a good idea would be to keep for training some statistics over all embedding vectors of each sentence. So, more than the [CLS] token, use the mean of all embedding vectors, the standard deviation, the maximum, the minimum...(actually we tried but it failed because of the memory of the machine).

VI. References

[1] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf - 2019 - *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter* - Hugging Face

[2] Victor Sanh - Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT - Medium

[3] Kaggle challenge : "https://www.kaggle.com/c/nlp-getting-started"

[4] 'Data For Everyone' : "https://appen.com/open-source-datasets/"

[5] Raymond Cheng - BERT Text Classification Using Pytorch - Medium

[6] Jay Alammar - The Illustrated Transformer

[7] Ryan - Learning PyTorch – Fine Tuning BERT For Sentiment Analysis

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin - 2017 - Attention Is All You Need

[9] Andrej Baranovskij - Fine-Tuning Hugging Face Model with Custom Dataset