Tweet Sentiment Extraction

Yu Zhong Duan (z5174781), Kan-Lin Lu (z3417618)

I. Introduction:

Language expresses sentiments through the unique structure and meaning behind, along with the increasing use of social media, sentiment analysis using machine learning algorithms has gain wide research and marketing attentions, considering the sheer volume of raw text data generated every second globally. As a sub field of Natural Language Processing (NLP), the vision for automated sentiment analysis is gathering public opinions and states of mind towards a given matter 'instantaneously', allowing more accurate decision making [1].

However, to further understand the process, instead of extracting the sentiment using given text, identify which phrases within the given text expresses the sentiment is the purpose of this study. The goal is to use a machine learning algorithm in picking out the phrases that reflects/supports the labelled sentiment. Table 1 illustrates an example of negative, positive and neutral texts, with respective selected text, given as training data.

Table 1 : Example of Sentiment Extraction

Raw Text	Sentiment	Selected Text
"oops too late"	Negative	"oops"
"Thanks"	Positive	"Thanks"
"Thx"	Neutral	"Thx"

The data given in this study is through Figure Eight's Data for Everyone platform, and are mainly Tweets from Twitters – spitted into train and test csv files.

Main challenges faced includes the irregular texts used in Tweets, including non-dictionary phrases, structureless sentences, irregular punctuations, emoticons, hyperlinks and informality constructions. Most importantly, local relationship between phrases.

The evaluation method is based on Jaccard Score, which measures dissimilarity between predicted selected text and ground truth selected text. Function is provided by competition organizer, and shown in *Implementation* section.

After several attempts, the final adopted method used in this project is enhanced version of BERT, a pre-trained model, named RoBERTa, details will not be discussed within this report, only implementation strategies and modification made. Further details and comparison of various BERT models is presented in [2].

II. Previous Attempted Methods:

This section aims to address the ML algorithm attempted and other popular methods that are conducted via competitors in the competition.

Bag of words with Naïve Bayes

Bag of words with Naïve Bayes was the intuitive algorithm that was initially taken by the team. The presence of irregular text and symbols were replaced and filtered, word frequency with respect to each sentiment was counted as an extra feature along with returned Vectors as input to the NB training model. Considering the spread of data observed in Figure 1, and the conducted Exploratory Data Analysis (EDA), the pseudo code of the approach is provided as below:

```
If Neutral:
    Select all words
Else:
    Select base on NB trained model
```

This is proven to be not a dependable choice in "predicting", in this case selecting text, due to heavy reliance on frequency of word occurrence and measure of known words [3] in finding respective sentiment. An illustration is shown Table 2.

Table 2 : Example of mislabelled sentiments.

Raw Text	Given Label	NB Sentiment Label
"ummm not with me?"	Neutral	Negative
"time to be nerd"	Negative	Neutral
"This is not good"	Negative	Positive

More specifically, the approach cannot take into account the connection between words within a single sentence. Take row three in Table 2 as an example, the occurrence of the word "good" is considered by the algorithm a positive count, but the word "not" does not reflect a negative sentiment. This in terms affected text selection process, hence after 2 weeks of improvement and seeking solution via traditional ML algorithm with local and global feature engineering, an alternative approach is sought by the team, considering regardless which traditional algorithm or feature chosen, limitation still exists due to complex nature of spoken language.

Note, an interesting approach using solely word count is observed in [4] with 0.65 Jaccard score accuracy, however the approach did not involved any sort of training, hence is not considered by the team, as it do not fit the given criteria of this project.

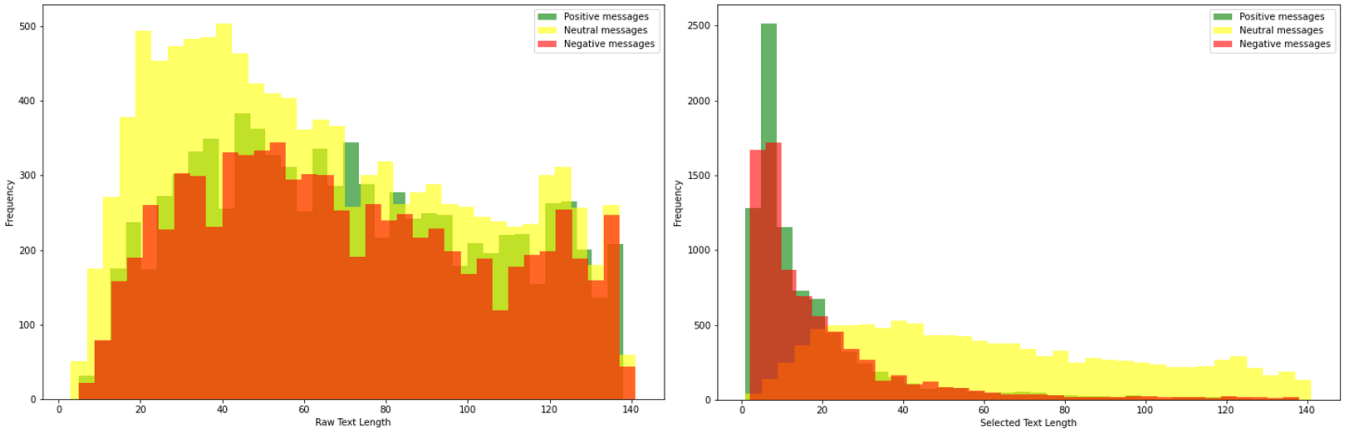

Figure 1 : (Left) Frequency against Raw Text length, (Right) Frequency against Selected Text Length.

BERT Model

The second approach the team adopted is using BERT pretrained module, considering the nature of the project this is a more valid approach. The obtained Jaccard Score, see Figure 2 for function, of ≈ 30% accuracy was obtained, which is an invalid outcome and mainly a consequence of large predicted "blank text" and nature of BERT model.

RoBERTa Model

Hence, after another week of BERT Model, the team draws attention to the popular algorithm that is taken by several different competitors with a high outcome accuracy: 70.5% - [5], 71.2% - [6], 71.4% - [7], 71.2% - [8] and 69.8% - [9].

Under the similar tokenization for RoBERTa model with huggingface [10], to the best of the team's understanding the difference between the presented models are:

Table 3 : Comparison of RoBERTa Approaches

Competitors	Approaches
[5]	MAX_LEN = 96, Train Batch Size = 32, Train Epoch = 3, K-fold = 5
[6]	MAX_LEN=192, Train Batch Size = 32, Train Epoch = 5, K-fold=5
[7]	Random Seed, MAX_LEN = 96, Train Batch Size = 8, Train Epoch = 3, K-fold = 10
[8]	MAX_LEN = 96, Train Batch Size = 8, Train Epoch = 5, K-fold= 5
[9]	MAX_LEN=192, Train Batch Size = 16, Train Epoch = 2

The team then begins investigate into the standard RoBERTa model for implementation. Some of the settings observed from above competitors are kept, and examined the effect on outcome.

III. Implementation:

This section aims to address evaluation method, Pre-processing (including tokenization) and the RoBERTa model setup, on the implementation aspect. Note, considering the level of sophistication of other competitors, the team aims to understand various approaches from the competitors and providing a reasonable set up for experimentation.

Evaluation

Jaccard Score dissimilarity function is provided as following:

```
def jaccard(str1, str2):
    a = set(str1.lower().split())
    b = set(str2.lower().split())
    if (len(a)=0 & (len(b)=0):
        return 0.5
    c = a.intersection(b)
    return float(len(c))/(len(a)+len(b)-len(c)
```

Figure 2 : Evaluation Method provided by [11].

Pre-Processing

The team first builds a dataset generator that translate the word in text to the matrix for RoBERTa module. Follow by that, a Question and Answer approach is appended on to RoBERTa pretrain module, without going into details of each component, the overall illustration is shown in Figure 3.

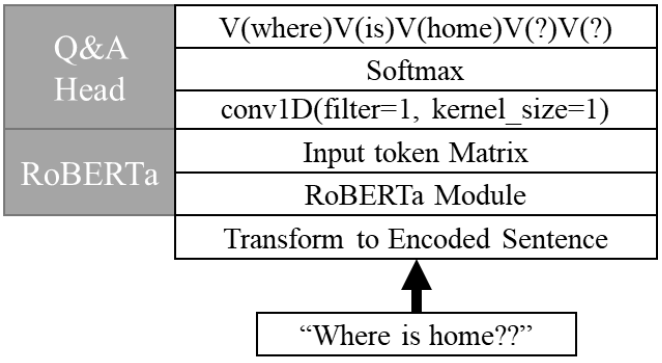

Figure 3 : Illustration of Pre-processing

To provide a brief description of the setup, the encoded sentence is transformed into an input token of [(32, 96, 768)] shape matrix. Follow by a Conv1D with 768 filters and with kernel_size of 1 in converting the encoded matrix to [(32, 96,1)] shape matrix.

Softmax is then use to flat the matrix to get a 2-D matrix, where x_1 represents the start token and x_2 represent end token. In order to cater model requirement, padding is then inserted.

An attribute error was encountered due to presence of a NULL value in given training data, however instead of deleting the entire row, as adopted by [12], the team replaced NaN with "", for the purpose of preserving the data.

Training model Parameters

Considering the competitors' approaches shown in Table 3 following configuration is set up:

- Max Word Length = 96
- Training Batch Size = 32
- Number of Epoch = 3

Note, Sentiment ID is based on RoBERTa built in vocab identification number, where positive, negative and neutral are 1313, 2430 and 7974 respectively, this follows similarly from the competitors. Label Smoothening of 0.1 is also introduced similarly to [13], with the purpose of reducing overfitting and overconfidence mentioned in [14].

The team then follows StratifiedKFold in providing cross-validation object, in ensuring higher accuracy for developed model. Weight is constantly saved and loaded through the defined save_weights, and load_weights functions via pickle, when model is being trained.

Loss Function

A loss function is introduced using Categorical Loss, similar to several competitors' approach, considering a multi-class classification nature of the model.

IV. Experimentation:

This section provides the experimentation description of described implementation. Experimentation is conducted via the given train.csv file. As a matter of fact, considering test.csv does not provide a ground truth comparison, the team focuses on using training data accuracy as an indicator.

Table 4 shows a sample result from 2-fold setting, as observed by the team, predicted text shown variance comparing to given truth. Hence a visual comparison is conducted and shown in Figure 4, where the team observed the main difference occurs for text below length of ≈ 35. Attempts in adjusting the model in

Table 4 : : Test.csv output Predicted Text (Selected 10 text for comparison purpose) base on 2 fold outcome.

Sentiment	*Raw_Text*	*Selected_Text*	*Predicted_Text*
Positive	lucky kid...i so wanna see loserville pity im in oz....	lucky	lucky kid...
Neutral	Certainly not Cheers than, huh?	Certainly not Cheers than, huh?	certainly not cheers than, huh?
Negative	those splinters look very painful...but you were being very heroic saving mr. Pickle	painful.	painful…
Negative	you said you **** up your nose!	**** up your nose!	****
Positive	Dunno yet, would LOVE to though! I keep missing them any other time for some reason :S & have fun!	would LOVE to though!	would love to though! i keep missing them any other time for some reason :s & have fun!
Positive	waking up at 6 am? yuuup. still going out tonight? you better believe it.	believe	yuuup. still going out tonight? you better believe it.
Positive	#snl...you`ve gotten better keep it up	you`ve gotten better	you`ve gotten better keep it up
Positive	Yes, we will! We should maybe do it in a DM though so we don`t annoy our followers. Thought of that too late. Good night!	so we don`t annoy our followers. Thought of that too late. Good nigh	good night!
Negative	had to many weirdos on the other one	weirdos	had to many weirdos
Negative	when I tried to go to her`s I got this message 'That page doesn`t exist! '	'That page doesn`t exist! '	doesn`t exist!

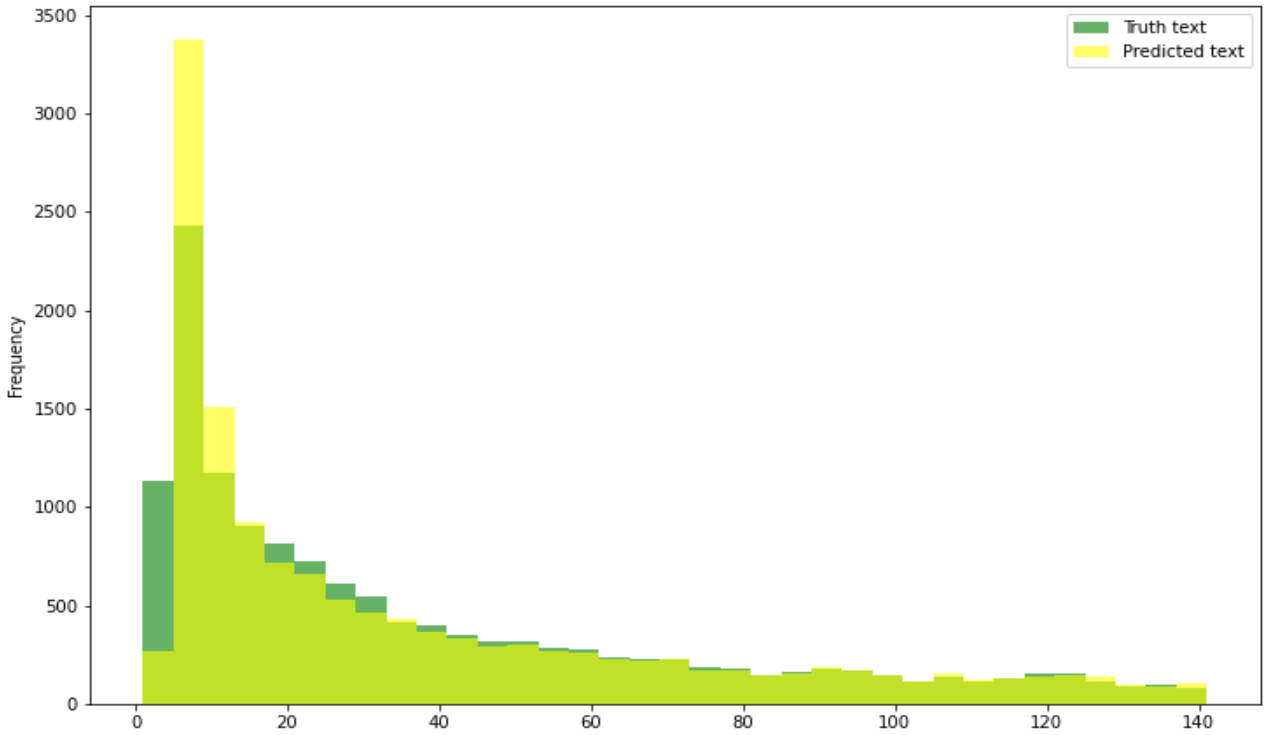

Figure 4 : Visual Comparison of Truth Text and Predicted Text base on 2 fold outcome.

specifically tackling this observation were not successful, hence the team draws the attention to number of folds adjustment and batch size.

Table 5 shows various K-fold experiment with respective Jaccard Score, see included result image for respective screenshots, 3-fold provides the highest accuracy outcome.

Table 5 : K-Fold Experiments.

K-Folds	Jaccard Score
2	0.7037
3	0.7049
4	0.7038
5	0.7045

To provide further comparison, a batch variation is also conducted, however in ensuring consistence and faster run time, number of folds is kept at 2. Observed from the team, batch size of 32 provided the highest accuracy.

Table 6 : Batch Size Experiments.

Batch Size	Jaccard Score
32	0.7037
64	0.7015

V. Results:

This section aims to address the obtained result, in this case the first five rows of produced submission.csv file, shown in Table 7.

Overall, the team is satisfied with the achieved result, considering the first few weeks of unsuccessful attempted using traditional machine learning methods with Bag of Words. Link and instruction to run the final selected method is provided in *Section VIII – Appendix*.

Table 7 : Sample outcome for submission.

textID	selected_text
f87dea47db	last session of the day
96d74cb729	exciting
eee518ae67	such a shame!
01082688c6	happy
33987a8ee5	i like

VI. Conclusion and Future Work

In conclusion, the project competition was successfully conducted and various methods, traditional and pre-trained models, were attempted by the team.

Considering the nature of the required task, that is selecting text from a given raw text with labelled sentiment, traditional approach lacks and unable in capturing the local relationship between words by simply using vectorization and feature engineering, at least through the attempts performed by the team.

In the end, in order to have a valid outcome, pre-trained RoBERTa model is chosen with described setting. However, the overall accuracy does not show an improvement comparing to the competitors' outcome, nevertheless considering the given time and limited team experience in natural language processing, this provided an insight into sentiment extraction.

In future, deeper understanding of overall RoBERTa model is definitely required, as observed from some competitors, dedicated pre-processing and post-processing results in higher precision.

VII. Reference:

[1] S. Gupta, "Sentiment Analysis: Concept, Analysis and Applications," *towards data science*, 2018. https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17.

[2] S. Khan, "BERT, RoBERTa, DistilBERT, XLNet: which one to use?," *KDnuggets*. https://www.kdnuggets.com/2019/09/bert-roberta-distilbert-xlnet-one-use.html.

[3] J. Brownlee, "A Gentle Introduction to the Bag-of-Words Model," 2017. https://machinelearningmastery.com/gentle-introduction-bag-words-model/#:~:text=A bag-of-words is,the presence of known words.

[4] N. Kopowrickz, "A simple solution using only word counts," [Online]. Available: https://www.kaggle.com/nkoprowicz/a-simple-solution-using-only-word-counts.

[5] C. Deotte, "TensorFlow roBERTa - [0.705]," 2020. https://www.kaggle.com/cdeotte/tensorflow-roberta-0-705.

[6] T. Abhishek, "roberta inference 5 folds," 2020. https://www.kaggle.com/abhishek/roberta-inference-5-folds.

[7] Sazuma, "Tweet Sentiment RoBERTa PyTorch," 2020. https://www.kaggle.com/shoheiazuma/tweet-sentiment-roberta-pytorch.

[8] K. Al-Kharba, "TensorFlow roBERTa - [0.712]," 2020. https://www.kaggle.com/al0kharba/tensorflow-roberta-0-712.

[9] C. Kang, "RoBERTa Baseline Starter (+ simple postprocessing)," 2020. https://www.kaggle.com/cheongwoongkang/roberta-baseline-starter-simple-postprocessing.

[10] "Tokenizers." https://github.com/huggingface/tokenizers.

[11] Kaggle, "Tweet Sentiment Extraction," 2020. https://www.kaggle.com/c/tweet-sentiment-extraction/overview/evaluation.

[12] Mr_KnowNothing, "Twitter sentiment Extaction-Analysis,EDA and Model." https://www.kaggle.com/tanulsingh077/twitter-sentiment-extaction-analysis-eda-and-model.

[13] W. H. Khoong, "[TSE2020] RoBERTa (CNN) & Random Seed Distribution." https://www.kaggle.com/khoongweihao/tse2020-roberta-cnn-random-seed-distribution.

[14] W. Wong, "What is Label Smoothing?," 2019. https://towardsdatascience.com/what-is-label-smoothing-108debd7ef06.

VIII. Appendix – Google Drive

The output model and respective result for selected RoBERTa method is given in:

Link

ReadMe file is provided in executing the code, along with sample outcomes. Please advised, it takes at least 30 minutes in full executing the code under 2 fold using Colab GPU.