Twitter Opinion Summarization & Trend Analysis
By - Jeet Paresh Mehta, Harsh Mishra

Datasets:

1) Extreme Summarization (XSum) Dataset. (https://www.tensorflow.org/datasets/catalog/xsum)

   There are two features: - document: Input news article. - summary: One sentence summary of the article.

2) Reddit Dataset. (https://www.tensorflow.org/datasets/catalog/reddit)
   Features include strings: author, body, normalizedBody, content, summary, subreddit, subreddit_id. We only use 2 of these features, Content and Summary. **Content** is used as 'document' and **summary** is used as 'summary'.

Both the datasets used are part of the Datasets module that can be imported using Tensorflow. Since the datasets are considerably large in size, they are not included as part of the submission. The code to download both of them can be found in our source code.

Source Code Description:

1) Extract_tweets folder.
   Contains python script to extract tweets with a given keyword (player/team names in our case) and the time period for which the tweets are to be extracted. The tweets extracted are written to a given csv and saved in the data folder.

2) Tweet_sentiment folder.
   Contains a python script which characterizes a tweet into Positive, Negative and Neutral. The sentiment analysis is done using the pre-trained model provided by cardiffnlp (https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment). The model is based upon Roberta and is finetuned on analysis with the TweetEval benchmark. Currently this script uses the csv files with the saved tweets and appends the sentiments as additional columns.

3) Keyword_extraction folder.
   Contains python scripts to extract keywords from a given tweet. The keybert (https://github.com/MaartenGr/KeyBERT) module is used to extract the keywords from the tweets. The code present in the submission uses the csv files saved using the Tweet_sentiment script and saves a new csv file based on the tweet sentiment containing the keywords.

4) Finetune_t5 folder
   Contains 2 scripts to finetune Google's t5 (small) model for summarisation task. The 2 scripts work for the Xsum and the Reddit dataset. The pre-trained t5 model was loaded from Huggingface.

5) Data folder
   Contains the csv files saved from the scripts mentioned in 1), 2) and 3).

Experiment Details:

2 datasets (XSum and Reddit) with their Golden summary provided were used to finetune Google's t5 model for summarisation task. Since our downstream task is to summarize Tweets, therefore we emphasized on choosing a dataset that involves *social media* data. As there were no datasets with tweets and their golden summaries, we chose the Reddit dataset to act as a proxy for the same. The reddit dataset consists of 3,848,330 posts, out of which we only took 10% of the data for fine tuning due to our compute restrictions. In order to map the training data to the expected test data, we chose posts consisting of at least 80 words and at max 560 words. An additional filter was put on the summary as well, posts with summaries in the range (280 <= word length >= 10) were chosen. 280 was chosen as the upper limit, as the expected summary of the test set (tweets) will ideally be not greater than the size of a single tweet. No such modifications were made on the XSum dataset.

Results:

1) **Extreme Summarization (XSum) Dataset.**

   Training Metrics:

   Training Data - 204045 rows, Max. input length = 1024, Max target length = 128

   batch_size = 4, Learning rate=2e-5, Weight decay=0.01, Number of training epochs=1

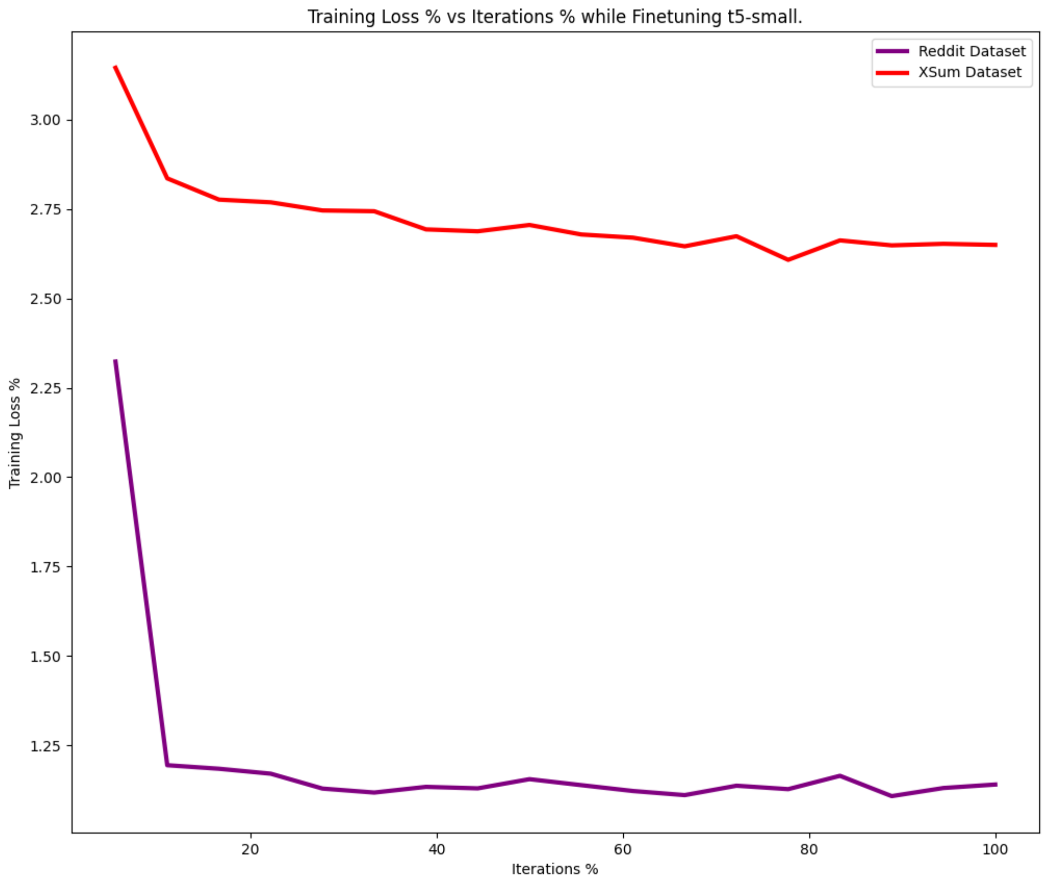
   Testing Metrics:

   Testing Data - 11334 rows, Loss: 2.419 %

   Rouge1 : 29.0994 (ROUGE-1 refers to the overlap of unigram (each word) between the system and reference summaries.)

   Rouge2 : 8.2844 (ROUGE-2 refers to the overlap of bigrams between the system and reference summaries.)

RougeL : 22.9664 (ROUGE-L: Longest common subsequence problem takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically.)



2) Reddit Dataset.

Training Metrics:

Training Data - 384,833 rows, Max. input length = 1024, Max target length = 280

batch_size = 4, Learning rate=2e-5, Weight decay=0.01, Number of training epochs=1

Testing Metrics:

Testing Data - 38,483 rows**,** Loss: 1.073 %

Rouge1 : 15.7252 (ROUGE-1 refers to the overlap of unigram (each word) between the system and reference summaries.)

Rouge2 : 2.906 (ROUGE-2 refers to the overlap of bigrams between the system and reference summaries.)

RougeL : 12.9949 (ROUGE-L: Longest common subsequence problem takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically.)

The command line outputs of the training and evaluation stage for both the models can be found inside their respective folders inside the Finetune_t5 folder.

Flow of Downstream tasks / Using the model:

1) The jupyter notebooks used to train the models have code blocks which can directly be used to test the summarisation capabilities of the model on any text input.
2) The other way of utilizing the model is to provide keywords and start/end time as search parameters for tweets. The *“Extract_tweets”* code will be utilized to extract those tweets. The *“Tweet_sentiment”* code will be utilized to analyze the sentiments of those tweets and segregate them based on those sentiments. The segregated tweets will then be provided as the input for the model to get the Abstractive summary. The *“Keyword_extraction”* code can finally be utilized to act as a validation proxy for the generated summary. The more the number of keywords the generated summary contains, the better the summary. This helps us compare the 2 different models, one of which is trained using a dataset summarizing news articles, whereas the other one is trained using a dataset summarizing reddits posts.

All the latest source code and updated results can be found on Github: https://github.com/harshm16/NLP_project.