DATS6312 Natural Language Processing

Final Project Report

Sentiment Analysis in Social Media

Yue Li, Aihan Liu, Shuting Cai

Advisor: Amir Jafari

Fall 2022

Dec 12, 2022

Table of Contents

Introduction

Reddit is a platform for users to spontaneously discover and share content that its categories on the site are known as “subreddits”, which is like a community organized for those people who have similar interests on the subreddits, while other audiences can vote or comment to show their opinions on the submitted post. Both the negative and positive of the text evaluation provide more details and information rather than just the number of likes or dislikes. The analysis of positive and negative evaluations is the main application scenario of sentiment analysis, with the objective of identifying and classifying the sentiments in the texts varies with the situation. In this case, the text evaluation would be gathered from two main subreddits: Windows and Mac OS. Three classes of "Positive," "Negative," and "Neutral" are taken into account in the sentiments. Using the pre-trained model of Twitter Roberta base sentiment and distilling BERT base trained on IMDB data, the sentiment would be identified with the score of three emotion classes, and with the application of integrated gradient, the sentiments of the word or sequence with the importance on the predicted sentiment would be determined.

Download and Overview of Dataset

The data can be extracted using Reddit API, sourced from Pushshift[1]. Since the Pushshift API limits the number of 1000 data per request, it is necessary to apply multi-threaded downloads. PMAW (Pushshift Multithread API Wrapper) [2] is a wrapper for the Pushshift API which uses multithreading to retrieve Reddit comments and submissions.

```
def get_pushshift(subreddit, limit, before, after, filter_attr, datat_type):
    api = PushshiftAPI()
    if datat_type =='submission':
        submission_result= api.search_submissions(subreddit=subreddit, limit=limit, before=before, after=after, mem_safe=True, filter=filter_attr)
        print(f'Retrieved {len(submission_result)} posts from Pushshift')
        submission_df = pd.DataFrame(submission_result)
        return submission_df
    if datat_type == 'comment':
        # return the comments that match the submission id
        comment_result = api.search_comments(subreddit=subreddit, limit=limit, before=before, after=after, mem_safe=True)
        print(f'Retrieved {len(comment_result)} comments from Pushshift')
        comment_df = pd.DataFrame(comment_result)
        return comment_df
# cite: https://pypi.org/project/pmaw/
# https://medium.com/swlh/how-to-scrape-large-amounts-of-reddit-data-using-pushshift-1d33bde9286
```

Figure 1. Data Download

There are nearly 100k observations from January 2021 through the end of October 2022. Some data preprocessing methods are applied to the dataset for EDA, including changing all words to lowercase, tokenization, removing stopwords, and lemmatization using the NLTK package.

Digging into the data, the percentage of data showing real incidents of MacOS and Windows in the submission is 0.50 while the ratio of data showing real incidents MacOS and Windows is 0.55:0.45.

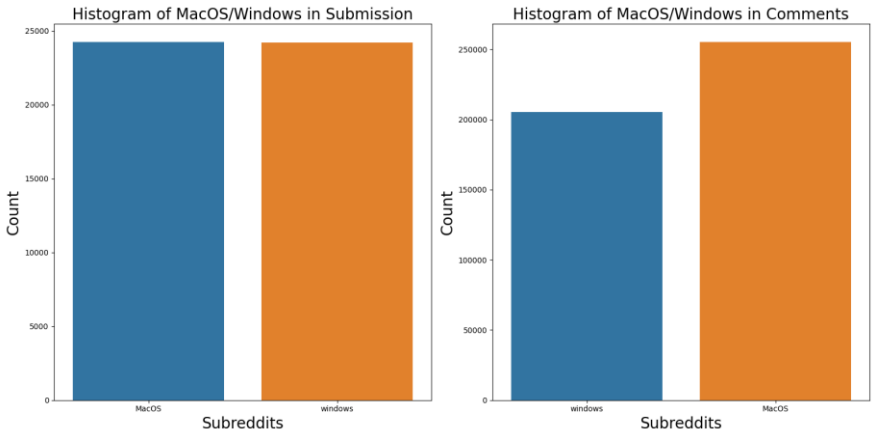

Figure 2. Histogram of the dataset for subreddits.

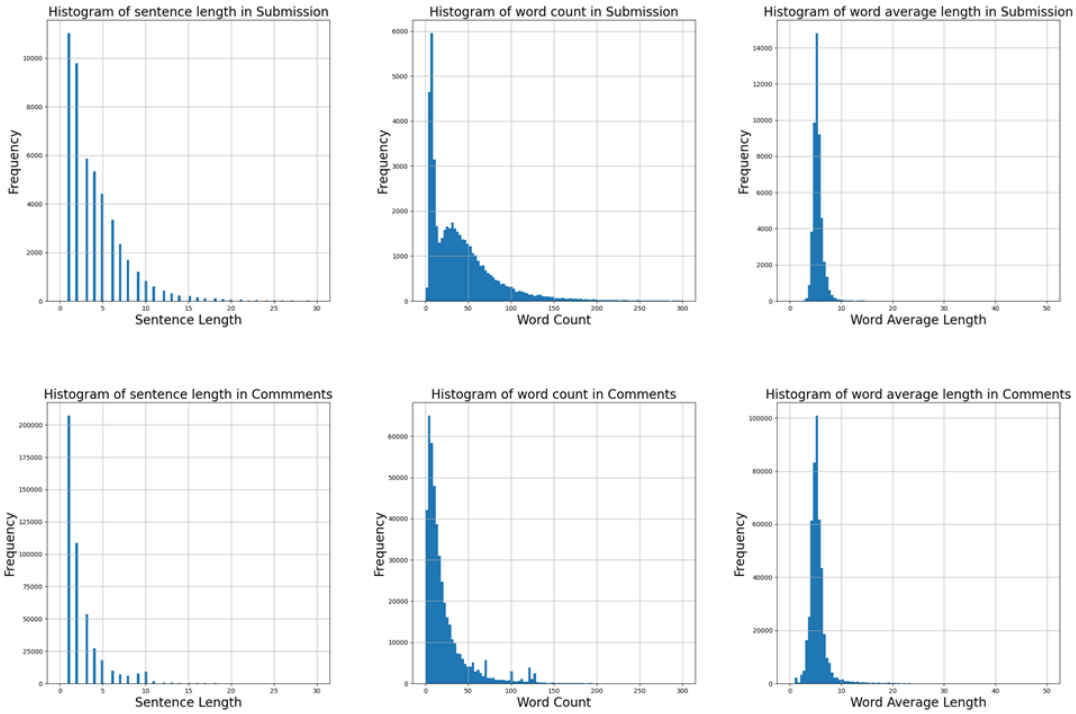

Figure 3. Histogram of sentence length, word count, and word average per sentence in Submission and Comments.

From the above figure, with a basic idea about the submission and comments before starting sentiment analysis. it is indicated that the sentence length and word count in the submission is higher than the comments. Most often, the submission person, known as a blogger needs more sentences and words to describe a thing he/she wants to talk about and share ideas, the viewer uses one word or one sentence to leave comments. The average word for each sentence is not a difference between submission and comments. In general, a normal sentence is 20-25 words,

considering it being a social media platform, people may be willing to express their ideas with a few words or abbreviations.

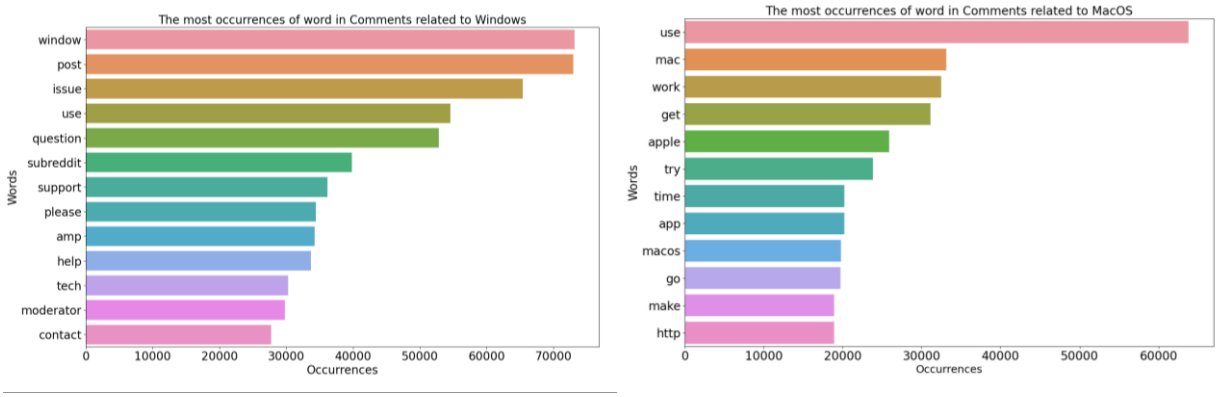

Figure 4. Bar plot of the most common words for Windows and MacOS.

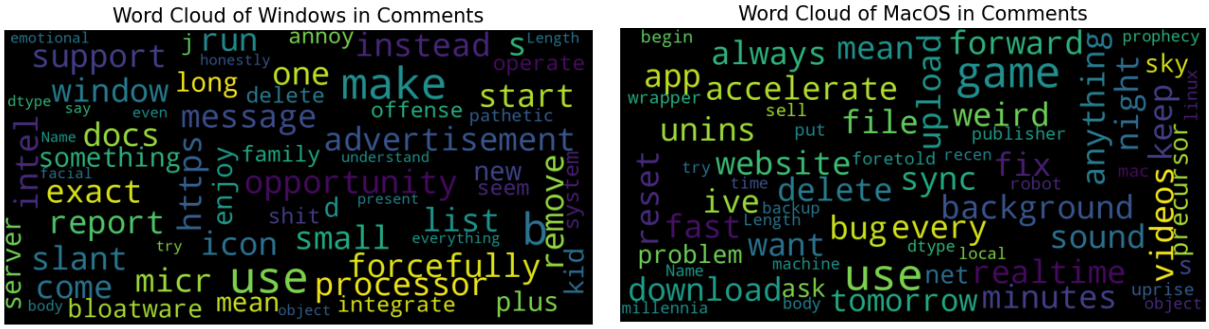

Figure 5. Word cloud for Windows and MacOS.

Checking with the most common word in subreddits, it can help people know the characteristics of Windows and MacOS and help Microsoft and Apple understand the demand and feedback of the user in these two operating systems. There are a couple of keywords in the bar plot and the word cloud that “please, ” “help, ” “issue,” “support” from windows users and “apple, ” “time,” “try,” and “weird” that conclude the commenter meet some problem when they are using the windows and MacOS and need help to solve it.

Model Descriptions

BERT stands for Bidirectional Encoder Representations from Transformers. BERT’s state-of-the-art performance is based on two things. First, novel pre-training tasks are called Masked Language Model(MLM) and Next Sentence Prediction (NSP). Second, a lot of data and computing power to train BERT. The MLM pre-training task converts the text into tokens and uses the token representation as an input and output for the training. A random subset of the tokens (15%) is masked, i.e. hidden during the training, and the objective function is to predict the correct identities of the tokens. The NSP task allows BERT to learn relationships between sentences by predicting if the next sentence in a pair is the true next or not. (reference)

RoBERTa was introduced at Facebook, Robustly optimized BERT approach RoBERTa, is a retraining of BERT with improved training methodology, 1000% more data, and compute power. To improve the training procedure, RoBERTa removes the Next Sentence Prediction (NSP) task from BERT's pre-training and introduces dynamic masking so that the masked token changes during the training epochs. (reference)

DistilBERT learns a distilled version of BERT, retaining 97% performance but using only half the number of parameters. The idea is that once a large neural network has been trained, its full output distributions can be approximated using a smaller network.(reference)

In this project, we selected to use a pre-trained RoBERTa-based model on a Twitter dataset and finetune a DistilBERT-based model on the IMDB dataset. The first model is named Twitter-RoBERTa-base for Sentiment Analysis (*cardiffnlp/twitter-roberta-base-sentiment*)[7]. It is trained on ~58M tweets and finetuned for sentiment analysis with the TweetEval benchmark. The reason to select this pre-trained model is the RoBERT-based model has the best performance among the other models mentioned above and it is trained on the Twitter dataset. It is a social media data set that matches the Reddit data set in this project. The second model is the DistilBERT-based model named distilbert-base-uncased. This model is a distilled version of the BERT-based model. It is uncased which means this model is not case-sensitive. This model is smaller and faster than BERT. In this project, the IMDB data set was selected to fine-tune from the hugging face website. IMDB data set contains movie reviews and binary sentiment labels. The model has been saved in hugging face and can access with the name: *laihanel/sentiment-analysis_gwu*

Experimental Setup

The DistilBERT model is trained with all the IMDB training data from the dataset package. The training loop is performed under the parameters shown in Table 1.

The results were outstanding: we achieved 89% accuracy on the test set (IMDB testing). On one hand, it proves our model architecture was correct and useful, on the other we were concerned with overfitting and model robustness.

In addition to fine-tuning a new model, there are over 1000 models in hugging face for the sentiment analysis task. We picked one of the most downloaded models (Twitter-RoBERTa-base model) to verify and compare the output with our trained model.

```
model = TFAutoModelForSequenceClassification.from_pretrained(model_name)
tokenizer = AutoTokenizer.from_pretrained(model_name)
```

Table 1. Training Parameters.

Parameter	Value
Epochs	1
Batch Size	16
tokenizer	Same as model
Maximum Input Length	256
Learning Rate	Linear learning schedules
Loss Function	SparseCategoricalCrossentropy
Optimizer	Adam
Evaluation Metrics	SparseCategoricalAccuracy

The team used these models to test the outcome of Reddit data. Since the Reddit data doesn't have any label for sentiment, we use the integrated gradients method to explain the relationship between model prediction and input characteristics.

Interpretability and Explainability

In the Interpretability and Explainability aspect, the team explores the integrated gradients (IG) method. In this project, a pertained transformer and fine-tuning model is used to do the sentiment analysis on Reddit submissions and comments text corpus. Our project focused on the posts under the categories of MacOS and Windows users. In these two categories, the sentimental analysis of posts is classified as positive, neutral, and negative. The sentiment analysis results came from the transformer model. The architecture of the transformer model has explained above. It is an attention neural network. It is hard to explain the results by only knowing the neural network's architecture. There are a lot of parameters connecting the input and output.

The point is how more explainability of this deep neural network model can be obtained. After getting the results from the model, the team tried to solve the problem of attributing the prediction of a deep neural network to its input features. In natural language processing, the attribution of a text sentiment network's prediction could be words or sentences. If the tokenizer is word-wise, the attribution would be the words, just like in this project. If the tokenizer is sentence-wise, the attribution would be the sentences. The importance of each word or sentence could give the output

of the neural network an explanation. Therefore, the interpretation and explanation are a reductive formulation of why this prediction is useful.

The gradient of the output with respect to the input is a natural analog of the model coefficients for a deep network. Therefore the product of the gradient and feature values is a reasonable starting point for an attribution method. In the computer vision domain, we can calculate the gradient of each patch with respect to the output. A black patch is used to cover an area in the picture. Then the difference in results between those with this black patch and without this black patch shows the importance of the covered area. For example, Δx is a pixel, Δy is the change of result. Then $\frac{\Delta x}{\Delta y}$ shows the importance of this pixel. y_k is the predicted label of an image and x_n is the n-th pixel. $\frac{\partial x}{\partial y}$ is the gradient. This is a typical gradient-based method. One issue of this method is gradient saturation. For example, the Sigmoid function would be saturating when x increases. Integrated gradients can solve this gradient saturation problem.

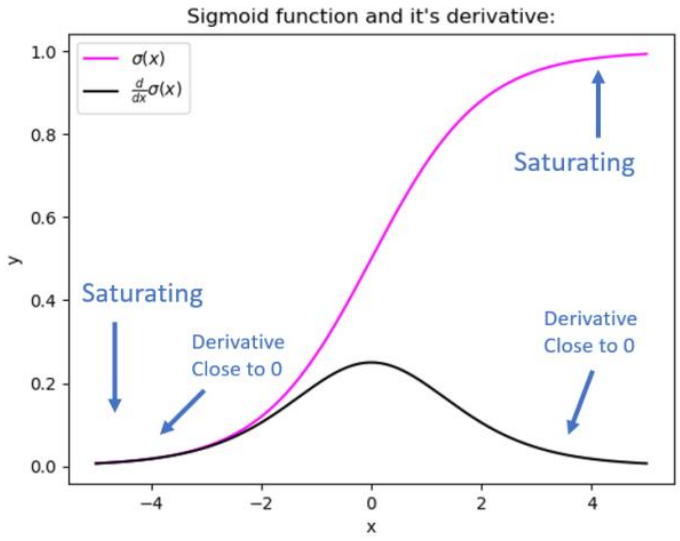

Figure 6. Example of sigmoid function and its first derivative.

The integrated gradient method was introduced to explain the neural network in a paper named Axiomatic attribution for deep networks by Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Two fundamental axioms guide the design of this method. One is sensitivity and the other is implementation invariance. Sensitivity means If for every input and baseline that differ in one feature but have different predictions, then the differing feature should be given a non-zero attribution. Implementation invariance means two networks are functionally equivalent if their outputs are equal for all inputs, despite having very different implementations.

Before implementing an integrated gradient method, it requires a baseline. The baseline is an informationless input for the model. For example, a black image for image models, empty text, or zero embedding vector for text models. Integrated gradients explain the network result of certain value input and the network result of baseline in terms of input features. The equation of the

integrated gradients method is below. Instead of calculating the direct gradient of a certain point, the integrated gradients method calculates the integration of a certain input value to the baseline.

$$\phi_i^{IG}(f, \mathbf{x}, \mathbf{x}') = \int_0^1 \frac{\delta f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))}{\delta x_i} d\alpha (x_i - x_i') \\ = (x_i - x_i') \int_0^1 \frac{\delta f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))}{\delta x_i} d\alpha \quad (1)$$

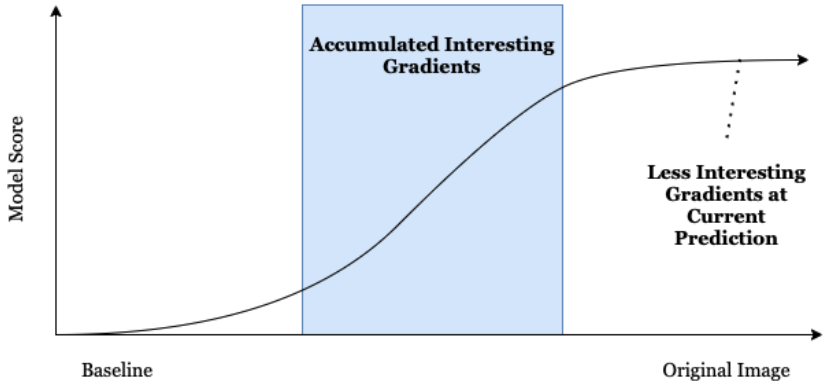

Figure 7. Example of Integrated Gradients for a Deep Learning Model.

Implementation and Results

The alibi implementation of the integrated gradients method is specific to TensorFlow and Keras models[8]. In this experiment, the embedding layer is involved in the gradient calculation as a baseline, which means we ensure that the attribution for tokens will be 0 if we use the embedding layer. The 0 attribution is due to integration between which is 0.

```
# calculate the attributions with respect to the
# first embedding layer of the (distil)BERT
elif model_name == 'distilbert-base-uncased':
    layer = auto_model.layers[0].layers[0].embeddings
# define Integrated Gradients explainer
ig = IntegratedGradients(auto_model,
                         layer=layer,
                         n_steps=n_steps,
                         method=method,
                         internal_batch_size=internal_batch_size)
```

```
# get explanation
explanation = ig.explain(X_test,
                         forward_kwargs=kwargs,
                         baselines=baselines,
                         target=predictions)

# Get attributions values from the explanation object
attrs = explanation.attributions[0]
print('Attributions shape:', attrs.shape)
```

Figure 8. Usage of IntegratedGradients

In Figure 9 and Figure 10, the sentiment of the sentence has been predicted from RoBERTa base model and the Distilling Bert base model. The words highlighted in green show us which words led the model to positive sentiment while the red words contributed to negative sentiment, the importance of word get from the different shades of color.

RoBERT base model is better than Distilling Bert base model because compared with movie reviews, the comments from social media are more similar to our data. In the meantime, the model performance is different too. Since we use the same model in tokenization as the training model, they do not have the same results in their tokens.

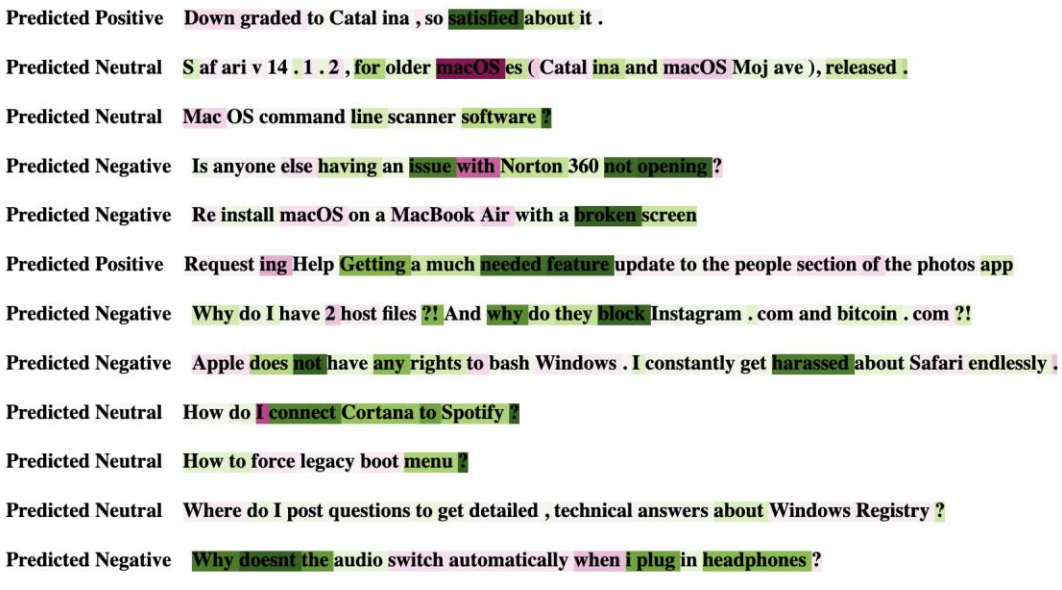

Figure 9. Predicted sentiments from RoBERTa-base model.

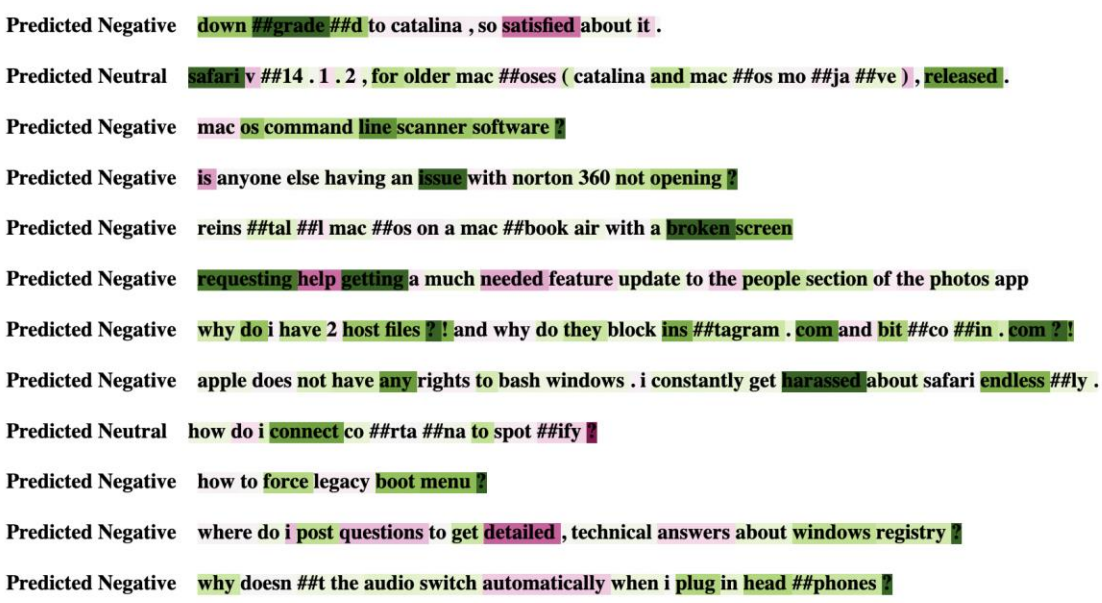

Figure 10. Predicted Sentiments from Distilling BERT base model.

Limitations and Future Work

This study is not perfect and is more of an attempt at a new field that we are not familiar with. There are a few points that were not improved due to time issues.

- As mentioned above, the tokenization method did not match, it probably affects the output of IG results.
- Due to a large amount of data and relatively long training time, the model was trained after only one epoch, and although it achieved good results on the testing set, it was not the best though.
- Methods for the integral approximation: we only applied a gausslegendre method, there are four more methods we haven't tried.
- By comparing the two we can see that the RoBERTa-base model gives more reasonable results. However because the training sets are different, it does not completely indicate that the RoBERTa model is better than the Distilling BERT. The difference in the focus of the two models has also been explained in Model Descriptions. In the future, if we can download the Twitter data and train different models, we can better explain whether there is a difference between these two models in using tweet data fine-tuning to Reddit.
- The study did not address well the question we posed: what are the factors that affect the quality of both windows and macOS subreddits reviews and the difference in user perception? A deeper study of the IG results could better answer the questions raised above.

References

[1] Baumgartner, J., Zannettou, S., Keegan, B., Squire, M., & Blackburn, J. (2020, May). The pushshift reddit dataset. In *Proceedings of the international AAAI conference on web and social media* (Vol. 14, pp. 830-839).

[2] PMAW: https://github.com/mattpodolak/pmaw

[3] Sundararajan, M., Taly, A., & Yan, Q. (2017, July). Axiomatic attribution for deep networks. In *International conference on machine learning* (pp. 3319-3328). PMLR.

[4] Sentiment analysis: https://huggingface.co/blog/sentiment-analysis-python

[5] Integrated Gradients: https://docs.seldon.io/projects/alibi/en/stable/examples/integrated_gradients_transformers.html

[6] Bert model comparision: https://towardsdatascience.com/bert-roberta-distilbert-xlnet-which-one-to-use-3d5ab82ba5f8

[7] Twitter-Roberta-base-sentiment: https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment

[8] alibi document: https://readthedocs.org/projects/alibi/downloads/pdf/stable/#page=99&zoom=100,96,609