PerSent Dataset Analysis Using Pretrained DistilBERT Model

Avish Parmar, Vincent Chi, Md Sadiq Sada
Stony Brook University, Stony Brook, New York
{avish.parmar, vincent.chi, mdsadiq.sada}@stonybrook.edu
Date: May 12, 2022

Abstract

NLP Project 1: Author Sentiment Prediction
News articles or opinions often express some sentiment towards an entity. In this project, we are building a system that can detect this.

1 Introduction

We are trying to develop a system that is capable of classifying articles as positive, negative, or neutral. This matters because news articles or opinions often express some sentiment towards an entity, and it is useful to extract that information in order to determine the purpose and rationality behind article pieces. This problem is difficult to solve given the size of news articles in which each paragraph receives a sentiment and then a classification technique is applied, such as majority voting, to determine the true sentiment of the article.

We need a standard language model based system (BERT, DistilBERT). Out of these models, we can use pretrained DistilBERT in order to save us from the large computation required to train our large models. A limitation in the above approaches is the maximum position embeddings. As DistilBERT is a small and lighter learning model, it defaults to a 512 maximum position embedding size. This presents an issue in the amount of data dropped when trying to learn.
The three papers we found relevant to this problem are the following:

1. Author's Sentiment Prediction: The main idea of this paper is to infer the sentiment of an author towards an entity based only on the text of their news article

2. Sentiment Analysis in News: The main idea of this paper is to present work on mining opinions about entities in English language news, in which (a) the authors tests the relative suitability of various sentiment dictionaries and (b) the authors attempt to separate positive or negative opinion from good or bad news.

3. Longformer: The Long Document Transformer: The main idea of this paper is to address the shortcomings of Transformer-based models which are unable to process long sequences due to their self-attention operation, which scales quadratically with the sequence length. The paper introduces the Longformer with an attention mechanism that scales linearly with sequence length, making it easy to process documents of thousands of tokens or longer.

We are addressing the classification problem of learning author sentiments through using the DistilBERT model. This relates to the ideas discussed in the papers in the following ways:

1. Author's Sentiment Prediction: Our project was an advancement of this paper in the sense that we sought to achieve the goals stated within this paper by using DistilBERT, a model not used in the analysis of paper's problem statement.

2. Sentiment Analysis in News: Our project relates to the ideas discussed in this paper because the paper also conducts a sentiment analysis on a dataset made up of news articles (i.e., training and testing the models on a similar dataset as us). Furthermore, our project also serves to distinguish between good and bad news by producing a sentiment for the article and paving the way for producing the sentiment for the author to determine credibility.

3. Longformer: The Long Document Transformer: This ideas discussed in this paper relates to our project as it provides a way for us to analyze the shortcomings of DistilBERT in view of Longformer, a transformer that is much more well equipped and fine-tuned for producing sentiments on large documents.

To know if our model is learning or not, the baseline is defined as $1/c$ where c is the number of classes. If our F1 score is over $1/c$, it means our model is learning. In this case, our baseline consisted of 3 classes (positive, negative, or neutral) with a baseline metric of 33% (1/3).

We evaluated our ideas by training the corpus on a DistilBERT model through many epochs. Our corpus comes from the PerSent data-set, which is a corpus containing 5k documents and 38 paragraphs annotated on the author's sentiment on the main entity of the article. We only needed 1 baseline. Our baseline consisted of 3 classes (positive, negative, or neutral) with a baseline metric of 33% (1/3).

1. We implemented DistilBERT for the analysis of the PerSent corpus to determine author sentiment.

2. We added onto our homework 3 implementation of DistilBERT to incorporate features necessary for the analysis of PerSent data-sets. For example, we added more classes and epochs so that Homework 3 is capable of analyzing the data-set.

3. Our Results for 512 embedding size performs better than the Analysis 128 embedding size models in F1 scores all across the board. Our evaluation for frozen embeddings was 0.1803 points higher, our top 2 training was 0.1791 points higher, our top 4 training was 0.1709 points higher, and our all training was 0.1694 points higher.

4. Our analysis shows that 128 embedding size works worse for all our tests. This is because given the large size of the news article corpus, we are eliminating a larger portion of the corpus when producing a sentiment compared to 512 embedding size. Therefore, the model run under 128 embedding size would produce much more ill-informed sentiments compared to 512 embedding size.

5. Based on our work we conclude that investigating other models like LongFormer, pre-processing the data to only include documents with a clear sentiment might be promising avenues for future work.

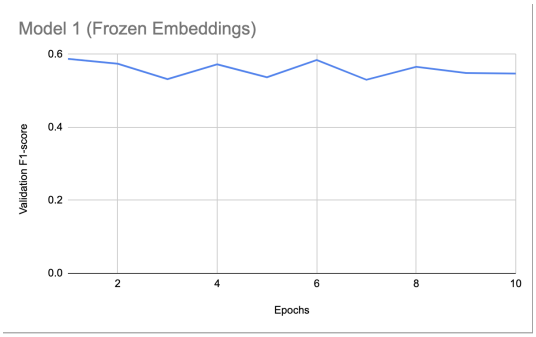
2 Your Task: Applying DistilBERT on PerSent dataset

The input for this task are news articles (documents) and the output is one of three possible classes: positive, negative, and neutral. The outputs indicate the author's sentiment over the original document. One of the key challenges with this task is that all of the sentences might not reflect the overall sentiment. We are tackling this problem with a simple approach - restricting the input to the first 512 tokens in the document as that is the limit for our model. The state of the art models used for this task include DAN, Modified Recurrent Entity Network, Hierarchical Discoursed based LSTM, and Large Pre-trained Transformer with BERT + Fine-tuning (Bastan et al., 2020). For this experiment, we are using a pre-trained Distilbert model as it is capable of completing different tasks including document classification. We experimented with different types of fine-tuning to see if the results made a significant difference. The 4 approaches were:

- Frozen embeddings: freezing the embedding layers
- Top 2 training: Only training the top 2 layers
- Top 4 training: Only training the top 4 layers
- All training: Training all 4 layers

3 Model 1: Frozen Embeddings

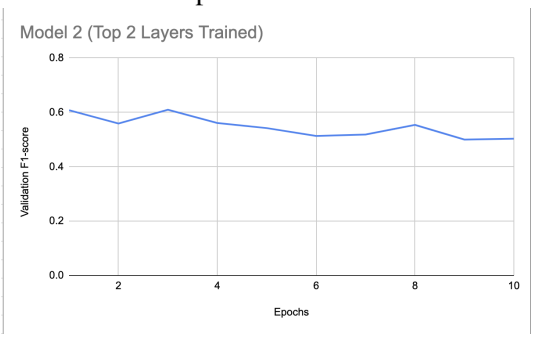
For the first approach, we froze the embedding layers of the distilbert-base-uncased model using the `requires_grad=False` statement. We trained the model for 10 epochs using the PerSent training data. The plot of the validation F1-scores are plotted below on a line chart:



The F1-score was somewhat consistent throughout the duration of the training phase. It varied between 0.5 and 0.6. The baseline for this task is 0.33, since we have three different classes, and the baseline performance is 1/c = 1/3 = 0.33. This model was performing better than the baseline model.

4 Model 2: Top 2 Training

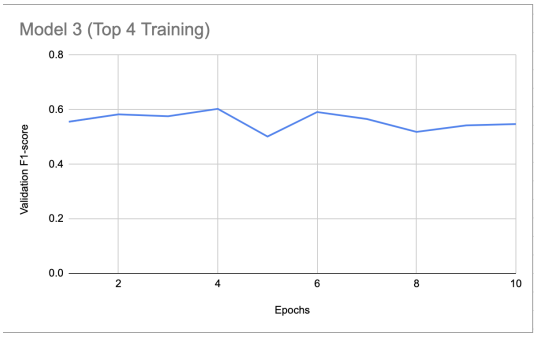
Distilbert has 6 transformer layers. For this approach, we froze the bottom 4 layers, along with the embeddings. Only the top two layers were trained. The validation F1 scores are plotted on a line-chart below:



The results are similar to the previous approaches. The model hits a plateau very quickly and the F1 score starts to drop. The value is hovering between 0.5 and 0.6, which is higher than our baseline performance of 0.33.

5 Model 3: Top 4 Training

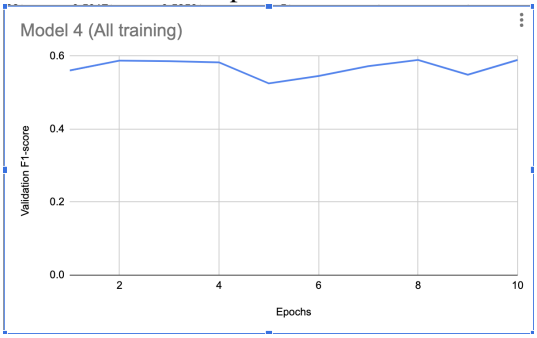
For this approach, we froze the bottom two transformer layers and trained the top four. The embedding layers were also frozen. The validation F1-scores are plotted on a line-chart below:



The scores are similar to our previous approaches. It increases for the first 4 epochs and then we see a sharp decline for epoch 5, and then it goes back up to the previous value and starts to plateau. The score is consistently close to 0.6. Our baseline was at 0.33. This model was also performing above the baseline from the get-go.

6 Model 4: All Training

This approach was similar to the previous ones, except instead of freezing some of the layers, we let all our layers be trained. The validation F1 scores are plotted below on a line chart:



Again, the values are similar to what we've seen before. The F1-score increases until epoch 2 and then plateaus for the rest of the graph. The value stayed between 0.55-0.6, which is above our baseline performance of 0.33.

7 Experimental Setup

The purpose of our evaluation is to test the DistilBERT model on a larger dataset such as PerSent. We analyze the precision, recall, and F1 measurements as they give us an indication of how the model is interpreting the training data and determine whether it is learning. The questions that we asked in evaluating the system for our task were:

1. Given that DistilBERT is a simpler version of BERT, how do it's results compare to those of BERT shown in the Author's Sentiment Prediction journal?

2. How does the model perform on a larger data set in comparison to the much smaller data set we had seen in Homework 3?

3. What were the major drawbacks of DistilBERT when applying it to the PerSent data set?

7.1 Dataset Details

The dataset is called PerSent, which is a corpus containing 5000 documents along with the authors' sentiment toward the content of the documents.

7.2 Evaluation Measures

We are using precision, recall, and F1 measurements to evaluate our results.
Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.
Recall is the ratio of correctly predicted positive observations to the all observations in actual class.
F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account.

7.3 Model Implementation Details

We used a pretrained model from Huggingface called distilbert-base-uncased.

7.4 Implementation Details

List the implementation details for the models (e.g. hyperparameters, what values did you set them to? why? how did you tune them?).
Loading the model: We used the from_pretrained function as described in hugging face documentation to load the distilbert-base-uncased model with three possible outputs.
Tokenizing the words: The tokenizer.encode_plus function takes in a string or a list of strings and converts them to token ids. We used the truncation parameter to ensure that the max length is capped at 512. We added the token list to the tokens array and the corresponding label to the labels array.
Iteration: We used a loop to iterate through model.named_parameters() and parsed the names to determine whether the current layer is an embedding layer or a transformer layer. Depending on what the input training_type was, we set the parameter tensor for the corresponding name to requires_grad = False to make the layers not trainable.
For the training step, we set the device for the reviews and labels tensor, and passed them into the pre-trained model. The output loss and logits were used to calculate the performance (precision, recall, f1 score). We used loss.backward() to back-propagate the loss.
For the validation step, we passed reviews and labels to the pre-trained model and got the performance. The only difference was that we didn't propagate the loss backwards.
For testing, we did the same things: we took the reviews and labels, and passed them into the model, and then passed the output loss and logits to the performance function. We didn't use loss.backward().
The hyperparameters used for this experiment were:

- batch size
- epochs

Initially, batch size was set to 16 and epochs was set to 3. Since this dataset is much larger, epochs=3 didn't give conclusive results. We increased the epochs variable to 10 to get more context for the results.

8 Results

We used the distilbert-base-uncased model for all of our tests. We experimented by fine-tuning the number of frozen layers and observed the impact on our results. The F1-scores were:

- Frozen Embeddings: 0.5670
- Top 2 Training: 0.5631
- Top 4 Training: 0.5681
- All Training: 0.5315

The scores were similar for all of the tests. It was almost the same for frozen embeddings, top 2 training and top 4 training. For all training, the scores was slightly worse. The performance was always above the baseline performance of 0.33. The reason it wasn't even better was possibly because of how Distilbert restricts tokenization to 512 words. We explore that more in-depth in the analysis section.

9 Analysis

We used distilbert-base-uncased to do the task of author sentiment analysis. Our F1 scores for the 4 different models were below 0.6. One hypothesis we had as to why our model wasn't performing better was that distilbert has a limitation on how many words it can tokenize. The limit is 512 words, and their are a lot of documents in our dataset with more than 500 words (Balahar el al., 2013). To test this hypothesis, we set the length of the tokenizer function encode_plus to 128. The F1-scores from the models were lower after we reduced the length. The new scores were:

- Frozen Embeddings: 0.3867 (31.8% lower)
- Top 2 Training: 0.3840 (31.8% lower)
- Top 4 Training: 0.3972 (30.08% lower)
- All Training: 0.3621 (31.88% lower)

These lower numbers justify our hypothesis that the length of the encoding affects the performance of our model. We could potentially get better results by using a model that allows for tokenizing of large documents, like Longformer (Beltagy et al., 2020).

10 Code

1. https://drive.google.com/drive/folders/1aHjL4KpvWNdkp_S8T0peUgeOYn8DQhSG?usp=sharing
2. The base code used for this project was from homework 3. The following changes were introduced for it to function appropriately for our needs. The constants of the base code were modified in which the number of epochs was set to 10, originally being 3. Since the PerSent model consists for three classes (positive, negative, neutral), the number of classes in model initialization was set to three. Inside of the DataLoader class, local variables within `tokenize_data` were modified to reflect the changes in the dataset. The local variables being `local_dict`, where neutral was added and defined with its respective numeric value, `review_list`, where the index was set to 'DOCUMENT' to reflect the correct column in the PerSent dataset, and `label_list`, where the index was set 'TRUE_SENTIMENT' to refer to the correct column in the PerSent dataset represenitng the sentiment for training.
3. The instructions to train and test the models are explained in the .ipynb file.
4. There are no system requirements as the model is being run on Google Colaboratory. All required libraries are downloaded and imported in their respective code cells.

11 Conclusions

In conclusion, to address the issue of classifying author sentiments, we decided to use the DistilBERT model because of its convenience and simplicity. We learned that learning models can perform well even when not given the ideal data-sets. For homework 3, we had an ideal data set appropriate for distilbert. We got our desired results within three epochs. This dataset was much more complex, with large documents and more sentiment classifications. Our F1-scores were comparatively lower because of that. As we explored possible ways to enhance the performance of our models, we learned about models like Longformer that can solve this issue with Distilbert. Some avenues for exploration from this project include:

- Using a different model like Longformer and comparing its results with DistilBERT to better understand the shortcomings of DistilBERT, and perhaps also analyze its strengths
- Preprocessing the data to only have good documents and good sentiments, where the documents clearly express an opinion
- Running it on models that were not explored on the PerSent paper, and comparing their performance

12 References

1. Balahur, Alexandra, et al. "Sentiment Analysis in the News." ArXiv.org, 24 Sept. 2013, https://arxiv.org/abs/1309.6202.
2. Bastan, Mohaddeseh, et al. "Author's Sentiment Prediction." ArXiv.org, 12 Nov. 2020, https://arxiv.org/abs/2011.06128.
3. Beltagy, Iz, et al. "Longformer: The Long-Document Transformer." ArXiv.org, 2 Dec. 2020, https://arxiv.org/abs/2004.05150.