

Human Language Technologies 2022-23

“Comparing Sentiment Analysis Performance of Two Models”



UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA

Professor:

Giuseppe Attardi

Candidate :

**Yanamandra Sharath Chandra
(585953)**

Abstract:

Sentiment analysis is a crucial task in natural language processing (NLP), aiming to identify and categorize the sentiment conveyed in textual data. It finds wide-ranging applications in various domains, including social media analysis, customer feedback assessment, and market research. In this report, we delve into two prevalent approaches for sentiment analysis: one harnessing the power of DistilBERT, a distilled version of the BERT model from the Transformers library, and the other utilizing the SentimentIntensityAnalyzer module from the nltk.sentiment library. We compare and contrast these approaches based on their methodologies, training data, contextual understanding, and performance, empowering researchers and practitioners to make well-informed decisions when selecting and applying them in specific use cases.

In our analysis, we opted to employ the DistilBERT model as a cutting-edge approach for sentiment analysis. DistilBERT offers comparable performance to the original BERT model while being significantly faster and more lightweight. By distilling the knowledge from the BERT model, DistilBERT achieves a streamlined architecture, making it an attractive choice for sentiment analysis tasks. We explore the training process, hyperparameter tuning, and evaluation using metrics such as F1-score to measure the performance of the DistilBERT model.

1. Introduction and Motivation:

The rise of social media platforms, online reviews, and user-generated content has led to an increased demand for analyzing and understanding sentiment expressed in textual data. Sentiment analysis, also known as opinion mining, has emerged as a crucial task within the field of natural language processing (NLP) to address this need. It involves automatically determining the sentiment or opinion conveyed in text, which holds significant importance for businesses and organizations seeking to gauge public sentiment towards their products, services, or brands.

In recent years, numerous techniques and models have been developed to tackle the challenges of sentiment analysis. Among them, two prominent approaches that have gained considerable attention are sentiment analysis using the DistilBERT model from the Transformers library and sentiment analysis using the SentimentIntensityAnalyzer from the nltk.sentiment module.

The DistilBERT (Bidirectional Encoder Representations from Transformers) model, based on transformer architecture, has revolutionized NLP tasks by capturing contextual information and understanding intricate linguistic relationships. Pre-trained on vast amounts of unlabeled data and then fine-tuned on sentiment analysis datasets, DistilBERT offers a powerful framework for sentiment analysis with state-of-the-art performance. Its ability to grasp nuanced sentiments, handle diverse data domains, and provide comprehensive contextual understanding makes it a compelling choice for sentiment analysis tasks.

In contrast, the SentimentIntensityAnalyzer adopts a rule-based approach. It relies on a pre-defined sentiment lexicon that assigns sentiment scores to words based on human-annotated sentiment values. By aggregating these scores, the SentimentIntensityAnalyzer calculates an overall sentiment score for a given text. This lightweight and efficient approach is interpretable and suitable for basic sentiment analysis tasks.

2. DistilBert Model vs SentimentIntensityAnalyzer from nltk

The main difference between sentiment analysis using the BERT model and SentimentIntensityAnalyzer from nltk.sentiment lies in the underlying approach and the training data used.

2.1. Model Architecture:

- distilBERT is distilled version of BERT(Bidirectional Encoder Representations from Transformers) is a transformer-based model that uses a deep neural network architecture with bidirectional attention to capture contextual information from the input text.
- SentimentIntensityAnalyzer is a rule-based model that uses a predefined set of lexical features and heuristics to determine the sentiment of a text.

2.2. Training Data:

- DistilBERT is pre-trained on large amounts of unlabeled text data from the internet using a masked language modeling objective. It is then fine-tuned on sentiment analysis datasets with labeled positive and negative examples.
- SentimentIntensityAnalyzer does not require explicit training. It uses a pre-defined sentiment lexicon that contains words with associated sentiment scores. The scores are based on human-annotated sentiment values.

2.3. Contextual Understanding:

- DistilBERT captures the contextual understanding of words and sentences by considering the entire input sequence and leveraging the surrounding words. It can model complex linguistic relationships and understand the nuances of language.
- SentimentIntensityAnalyzer does not consider the context of the entire text. It assigns sentiment scores to individual words based on their presence in the lexicon and aggregates them to calculate the overall sentiment of the text.

2.4. Performance:

- DistilBERT has achieved state-of-the-art performance on various NLP tasks, including sentiment analysis. It can handle more nuanced and complex sentiment expressions and perform well even on out-of-domain or unfamiliar data.
- SentimentIntensityAnalyzer performs well on general sentiment analysis tasks but may not capture subtle nuances or handle domain-specific sentiments as effectively as BERT.

3. Dataset and Preparation:

Social media data has become vital for scientific research, but challenges in data engineering often hinder progress. The Pushshift Reddit dataset offers a solution by granting real-time access to Reddit data since its inception. Researchers can utilize the Pushshift API and computational tools to search, aggregate, and perform exploratory analysis on the comprehensive dataset. This saves time on data collection, cleaning, and storage.

In the initial stage of the analysis, the text data from the Reddit dataset underwent several preprocessing steps. The process involved converting the text to lowercase, removing numbers and punctuations, tokenizing the text into individual words, eliminating stop words, lemmatizing the words to their base form, and ultimately joining the processed words together to obtain a suitable format for conducting sentiment analysis.

The dataset comprises an ingest engine, PostgreSQL database, Elastic Search document store cluster, and API. The collection process involves retrieving data from APIs, web scraping, and data streams.

Pushshift's computational resources can handle 500M requests per month, providing search and aggregation capabilities.

For analysis, we utilized the monthly dump of Reddit data from Pushshift, focusing on selected subreddits and the timeframe between 01/01/2018 and 31/12/2018. In my analysis I have 20000 rows for training and 3000 rows for test and validation. During the training and model design I did not touch the test data.

Initially I have data in format figure 3.1 which is having all the sentiment scores, text and the year.

	month_str	processed_text	positive_score	negative_score	neutral_score
0	2018-01	thats great quote great man nonetheless eaten ...	0.235566	0.121782	0.641505
1	2018-02	read jack sparrow voice fear quite useful tool...	0.244673	0.124182	0.629824
2	2018-03	might call gentle giant always wonder bird cho...	0.240777	0.117634	0.640117
3	2018-04	struggled depression pretty hard even tried en...	0.237326	0.128808	0.632118
4	2018-05	im stupid full doubt wait mean there skilling ...	0.240785	0.119211	0.637241
5	2018-06	shark likely lonely depressed bit dick every c...	0.235813	0.131941	0.630818
6	2018-07	become school teacher summer awesome another g...	0.240657	0.122012	0.635964
7	2018-08	rwowthanksimcured cant workout right supposed ...	0.240193	0.124873	0.626593
8	2018-09	honestly seems like disconnecting friend tryin...	0.237336	0.132584	0.627858
9	2018-10	forgive highly recommend looking kyle cease st...	0.241567	0.134168	0.622307
10	2018-11	damn guess isi right im sure german woman rape...	0.243668	0.130738	0.620286

Figure 3.1.

	month_str	processed_text	positive_score	negative_score	neutral_score	probs
0	2018-01	thats great quote great man nonetheless eaten ...	0.235566	0.121782	0.641505	[0.29586130380630493, 0.3892509639263153, 0.31...
1	2018-02	read jack sparrow voice fear quite useful tool...	0.244673	0.124182	0.629824	[0.30485212802886963, 0.3804483413696289, 0.31...
2	2018-03	might call gentle giant always wonder bird cho...	0.240777	0.117634	0.640117	[0.36798611283302307, 0.3610939383506775, 0.27...
3	2018-04	struggled depression pretty hard even tried en...	0.237326	0.128808	0.632118	[0.283939003944397, 0.3871860206127167, 0.3288...
4	2018-05	im stupid full doubt wait mean there skilling ...	0.240785	0.119211	0.637241	[0.2956819534301758, 0.38850587606430054, 0.31...

Figure 3.2

	text	sentiment_label	tokens
0	youll thank one day	2	{'input_ids': [101, 2017, 3363, 4067, 2028, 21...
1	heh mini snack baggies market little kid think...	2	{'input_ids': [101, 2002, 2232, 7163, 19782, 4...
2	love	0	{'input_ids': [101, 2293, 102], 'token_type_id...
3	thank guy much reached rape crisis hotline res...	2	{'input_ids': [101, 4067, 3124, 2172, 2584, 90...
4	house fire red cross brought u thing like toot...	2	{'input_ids': [101, 2160, 2543, 2417, 2892, 27...

Figure 3.3

Finally for out distilBERT model I used this formatted data in fig 3.3. where I tokenized all the text data and appended in to the original dataset to a make model to understand easy. In between we have a few missing values, I also handled them as well and the final dataset ready to split looks like in figure 3.3.

4. Model Architecture:

I performed everything on Google Colab for extra GPU power and really helped me a lot for faster calculations.

Model Architecture for DistilBERT:

The DistilBERT SentimentClassifier model is designed for sentiment analysis using a combination of the DistilBERT model and an LSTM (Long Short-Term Memory) layer. This architecture allows the model to capture both the contextual understanding from the transformer and the sequential information from the LSTM. The SentimentClassifier class is defined as a subclass of **nn.Module**, the base class for all neural network modules in PyTorch. It takes several Parameters, these parameters define the configuration of the model.

The model architecture consists of the following components:

1. DistilBERT Transformer: The transformer module is initialized using the DistilBertModel from the transformers library. The **pretrained_model_name** parameter specifies the name of the pre-trained DistilBERT model to use. This component is responsible for encoding the input text and extracting contextualized representations.

2. LSTM Layer: An LSTM layer is added after the transformer. The LSTM takes the **hidden_size = 763** as both the input size and hidden size, with batch_first=True. This layer processes the outputs from the transformer in a sequential manner, capturing the sequential information and dependencies within the text.

3. Dropout Layer: A dropout layer is introduced after the LSTM layer to mitigate overfitting. The dropout_rate parameter controls the probability of dropout, which randomly sets elements of the input to zero during training.

4. Fully Connected Layer: A linear layer (nn.Linear) is applied to the final hidden state of the LSTM. It maps the hidden state to the number of classes (num_classes) in the sentiment analysis task. The output of this layer represents the logits for each class.

During the forward pass, the inputs are passed through the transformer, and the hidden states are extracted. The LSTM layer processes these hidden states, and the final hidden state is obtained by squeezing the output. The dropout layer is applied to the hidden state, and the logits are computed by passing the dropout output through the linear layer.

The model is initialized with the specified hyperparameters, and the device is set to GPU (cuda). Moving the model to the device enables GPU acceleration for faster training and inference if a compatible GPU is available.

This architecture combines the power of the transformer-based DistilBERT model and the sequential information capturing capability of the LSTM, resulting in an effective model for sentiment analysis tasks.

5. Implementation and Training:

In the implementation step as the model architecture is ready we need to pass the data to the model in the way it understands. The DistilBert understands the data in the format of **tensors**. so we need to **encode the tokenized text** and convert into tensors and load it into the **dataloaders**.

The dataloaders are having a tensors of {input_data, attention_mask, labels} in dictionaries {key: value} pairs.

Now, we need to find the hyperparameters to train the model. I tried on various combinations of hyperparameters by adjusting the **batch size**, **number of epochs**, **Learning rate**, and changing the **optimizer**.

After performing the grid search the best hyperparameters are :

```
learning_rate = 0.0001
batch_size = 16
num_epochs = 5
```

Initially I tested it on learning rate of **0.001** but the results were not so quite good since as the model is not learning at this rate. I also adjusted the batch size to **8** but it is taking much more time and finally I have taken number of epochs equal to **5** since the model is performing well from epoch 4.

Using the hyperparameters mentioned above, I started training the model on GPU. It took a few hours to train and the results are quite good which will be explained on section 5 Results.

6. Results and graphs:

The training of the sentiment analysis model using the DistilBERT architecture was performed over 5 epochs. During each epoch, the loss, which measures the deviation between the predicted and actual labels, gradually decreased. This indicates that the model learned to make more accurate predictions as it was exposed to more training data. The final loss value of 0.0776 suggests that the model achieved a high level of accuracy in classifying sentiments.

The validation F1-score, which measures the model's overall performance on the validation set, started at 0.8859 in the first epoch and steadily increased to 0.8950 in the fifth epoch. This indicates that the model improved in its ability to correctly classify sentiments as positive, negative, or neutral over the training process. A higher F1-score indicates better performance, and the achieved score of 0.8950 suggests that the model is effective in sentiment classification.

The results indicate that the model successfully learned from the training data and generalized well to the validation set, as evidenced by the decreasing loss and increasing F1-score. These metrics suggest that the model is capable of accurately classifying sentiments in unseen data.

```
Epoch 1/5: Loss: 0.4033636321297714, Val F1-Score: 0.8859443882534721
Epoch 2/5: Loss: 0.2049401729239949, Val F1-Score: 0.8947587134433282
Epoch 3/5: Loss: 0.11445843069488183, Val F1-Score: 0.9057909193843998
Epoch 4/5: Loss: 0.08149987924917201, Val F1-Score: 0.8820809024397902
Epoch 5/5: Loss: 0.07761695404560305, Val F1-Score: 0.8950398208619635
```

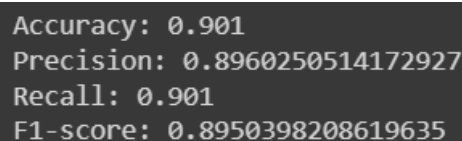
Figure 5.1

```
Test F1-Score: 0.8850140739617232
```

Finally, The sentiment analysis model was evaluated on an independent test set, and it achieved a test F1-score of 0.8850. This score indicates the model's ability to accurately classify sentiments on unseen data that it has not been previously exposed to. A higher F1-score suggests better performance in correctly identifying positive, negative, and neutral sentiments.

The achieved test F1-score of 0.8850 demonstrates that the model has successfully learned meaningful patterns and features from the training data, allowing it to generalize well to new examples. This result indicates that the model is reliable and effective in sentiment classification tasks, and it is likely to provide valuable insights when applied to real-world applications or scenarios.

It is important to note that the test F1-score serves as an objective evaluation metric and provides a comprehensive measure of the model's performance. This score reflects the balance between precision and recall, indicating both the model's ability to correctly identify positive and negative sentiments and its capability to avoid misclassifications. With a test F1-score of 0.8850, the model demonstrates a high level of accuracy and reliability in sentiment analysis.



```
Accuracy: 0.901
Precision: 0.8960250514172927
Recall: 0.901
F1-score: 0.8950398208619635
```

Figure5.3

The results from figure 5.3 of the sentiment analysis model on the test set are as follows:

- Accuracy: 0.901: This indicates the proportion of correctly classified sentiments out of all the predictions made by the model. An accuracy of 0.901 suggests that the model correctly classified sentiments in 90.1% of the cases.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

- Precision: 0.896: A precision of 0.896 suggests that the model correctly identified 89.6% of positive sentiments out of all the positive predictions it made.

$$\text{Precision} = TP / (TP + FP)$$

- Recall: 0.901: Recall, also known as sensitivity or true positive rate, A recall of 0.901 indicates that the model successfully identified 90.1% of positive sentiments out of all the actual positive instances.

$$\text{Recall} = TP / (TP + FN)$$

- F1-score: 0.895: An F1-score of 0.895 suggests that the model achieved a good balance between precision and recall, indicating strong overall performance in sentiment classification.

$$\text{F1-score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

Graphs:

