Explainability of Transformer Models in Sentiment Analysis and Comparison of the interpretability tools

Abstract

With so many popular Deep learning models being developed and solving complex issues, they still remain as black boxes. As a result, there is a untrustworthy notion towards them because of the lack of transparency. Many tools were created to work out the interpretability of such models however, there are few such examples in the domain of Natural Language Processing, more specifically, Transformer models. This experimental project aims to look at the interpretability of Transformer models using two recent tools (LIME and Captum) against a baseline CNN model on a Binary classification dataset. Finally the tools are compared to look at the different interpretability results.

1 Introduction

As the need for solutions to complex problem in computer vision and natural language processing continues to increase, so does the development of complex models to solve such problem. When dealing with simple models, like logistic regression, it is possible to work out exactly how the inputs are mapped to the outputs. However, as models get more complex with millions of trainable parameters, they become blackboxes, providing almost little to no explanations for their predictions with the human-readable explanations remaining absent. As a result, we can only see the input and the output and trust that the model is learning correctly. Tools were created to tackle these interpretability problems, however there is little such example in natural language processing and newly transformer models. Applying modern NLP for real-world applications requires interpretability and to make the system more robust and transperent. So, using established explainability tools such as lime and captum in various sentiment analysis datasets, we hope to bring some transperency into the NLP models with a degree of certainty and finally compare the two tools used for interpretability.

2 Literature Review

The works on BERT [1], LIME [2] and Captum (Integrated Gradients) [3][4] gives the necessary background on the tools and models required to work with this project. The work with sentiment analysis [5] looks at comparing two models and the tradeoff between them in terms of computation and accuracy of the explainability. It didn't dive deep into the datasets or the interpretability surrounding the dataset. The authors also built their own generative framework for interpretability and didn't use any tools. The work on Explaining Sentiment in the field of medicine [6] Looks into the need for explainable systems, compares different explainable systems, and brings up the need

for a better explainable system. This paper also does not deal with any datasets or any specific tools used.

3 Background

Before diving into the task, some high level understanding and intuition is presented in this section to show how the models and the tools work.

3.1 CNN

Usually Convolutional Neural Networks (CNN) are used for analyzing visual images. The basic architecture of CNNs is an alternating order of Convolution layers and pooling layers. Afterwards the feature matrix are flattened out into a vector and then followed by a Multi Layer Perceptron (MLP). Just like images can be represented as an array of pixel values, similarly text can be represented as an array of vectors that can be processed with the help of a CNN. When working with sequential data, like text, one dimensional convolutions are used, but the idea and the application stays the same.

The first task is to Vectorize a text corpus into a list of integers. Each integer maps to a value in a dictionary that encodes the entire corpus, with the keys in the dictionary being the vocabulary terms themselves.

each text sequence has a different length of words so padding with 0 is used to make sure every text has the same length.

Finally GloVe (Global Vectors for Word Representation) is used to get the word embeddings and an embedding matrix is created for the CNN.

3.2 Bert

BERT [1] is a multi-layer bidirectional Self-attention Transformer encoder based on the original implementation described in Vaswani et al. (2017) [7]. It gives an exhaustive background description of the model architecture as well as refer to excellent guides such as “The Annotated Transformer.” [8]. To give a high level overview of BERT, it is a series of encoder layers stacked on top of each other. The output layer of the model is modified according to specific tasks. For example, in the case of sentiment analysis, a feed forward network with a softmax layer is used in the output layer.

An encoder layer is a series of attention layer and a feed forward layer. In the attention layer, self attention works by allowing it to look at other positions in the input sequence for clues that can help lead to a better encoding for the word. The feed forward layer is a stack of Recurrent Neural Networks.

The BERT model used in this project is BERTBASE (Number of Layers=12, Hidden Size=768, self attention head=12, Total Parameters = 110M)

The model was pre-trained using the BooksCorpus (800M words) [9] and English Wikipedia (2,500M words). This pre-trained model is then trained on our custom dataset by only training the output layer of the model.

3.3 Lime

LIME (Local Interpretable Model-agnostic Explanations) [2] is a explanation technique that explains the prediction of any classifier by learning an interpretable model locally around the prediction. Locally interpretable means looking at Why did the model make a specific prediction and What effect did this specific feature value have on the prediction.

The output of LIME reflects the contribution of each feature to the prediction of a data sample. This provides local interpretability, thus showing us which changes in feature will have the most impact on the prediction.

An explanation is created by approximating the underlying model locally by an interpretable one. Interpretable models are e.g. linear models, decision tree's, etc. The interpretable models are trained on small slices of the original instance and should only provide a good local approximation. The 'dataset' is created by e.g. adding noise to continuous features, removing words or hiding parts of the image. By only approximating the black-box locally (in the neighborhood of the data sample) the task is significantly simplified.

Dense embeddings on the other hand are not interpretable, so applying LIME probably won't improve interpretability of such models.

3.4 Integrated Gradients

Formally, suppose we have a function $F : R^n \rightarrow [0, 1]$ that represents a deep network. Specifically, let $x \in R^n$ be the input at hand, and $x' \in R^n$ be the baseline input. For image networks, the baseline could be the black image, while for text models it could be the zero embedding vector. We consider the straightline path (in R^n) from the baseline x' to the input x, and compute the gradients at all points along the path. Integrated gradients are obtained by cumulating these gradients. Specifically, integrated gradients are defined as the path intergral of the gradients along the straightline path from the baseline x' to the input x.

The integrated gradient along the i^th dimension for an input x and baseline x' is defined as follows. Here, $\partial F(x) \partial x_i$ is the gradient of F(x) along the ith dimension.

$\text{IntegratedGrads}_i(x) ::= (x_i - x'_i) \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha$

4 Methodology

The dataset picked for this task is the restaurant reviews dataset and the Stanford Tree bank dataset. Both of them are binary classification datasets. The dataset structure is simple, containing only the texts and corresponding sentiments. For preprocessing, it is made sure that the classes are not imbalanced and this is resolved by downsampling the majority class. Afterwards the dataset is trained on a pre-trained Bert model and a CNN model. The CNN model is a baseline against which the Bert model will be compared. The CNN model converts the texts embeddings . The Bert model is trained by using a tokenizer specific to Bert. The tokenizer enocdes the text into input ids, attentions masks and position ids. These then act as an input to the bert model which then outputs a value between zero and one. The loss -Total loss as the sum of the masked language modeling loss and the next sequence prediction (classification) loss- is then calculated and weights are updated. The model is then ready to be shipped to Lime and captum to see how it predicts on specific sentences from the dataset.

5 Results

For the integrated gradients results, the labels are given. For the LIME results, blue means towards positive and orange mean towards negative

5.1 Restaurant reviews

5.1.1 CNN

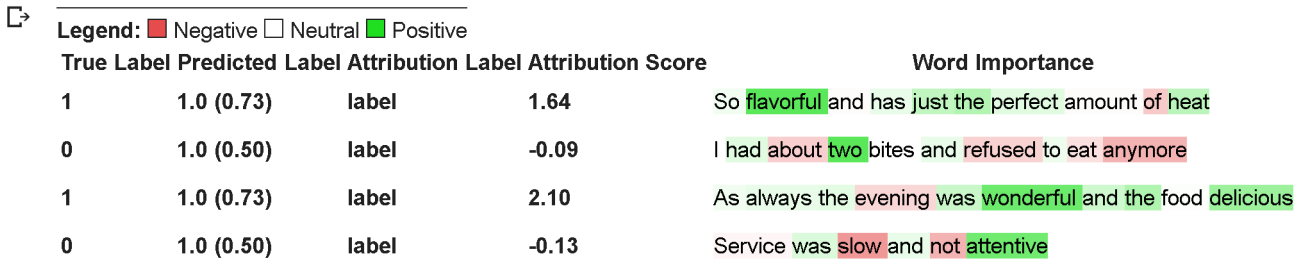

Figure 1: Result of integrated gradients on CNN

Figure 2: Result of Lime on CNN

As it can be seen for CNN, the highlights from the integrated gradients make sense and is consistent throughout, some of the results from LIME matches with that of the integrated gradients but LIME overall is bit more inconsistent.

5.1.2 BERT

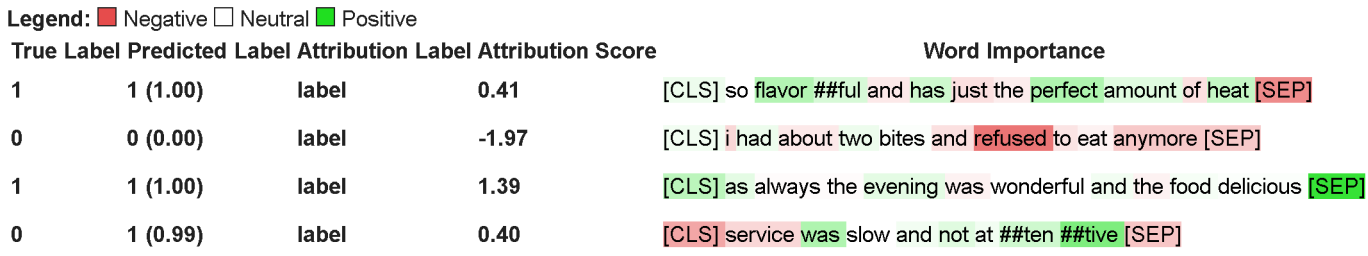

Figure 3: Result of integrated gradients on BERT

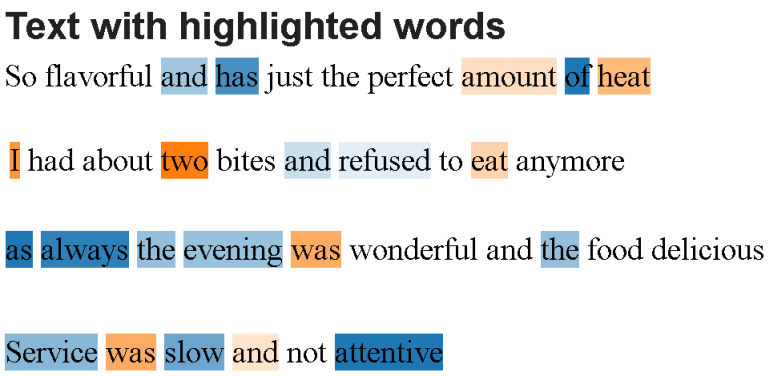

Figure 4: Result of Lime on BERT

For BERT, the results for integrated gradients also make sense but it seems less 'confident' given by the intensity of the colors. For LIME, the results are a bit inconsistent as words like slow and refused were given a positive connotation.

5.2 Stanford Tree Bank

5.2.1 CNN

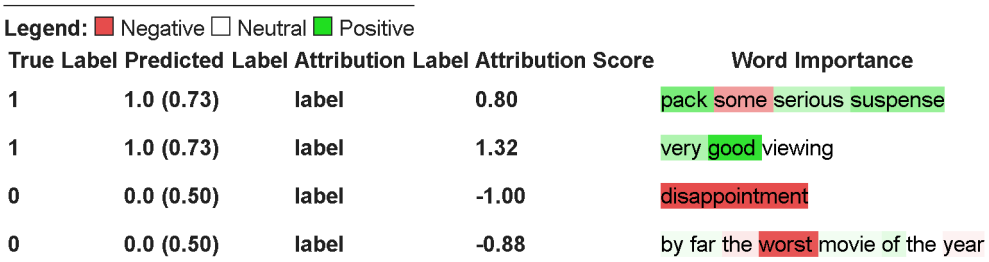

Figure 5: Result of integrated gradients on CNN

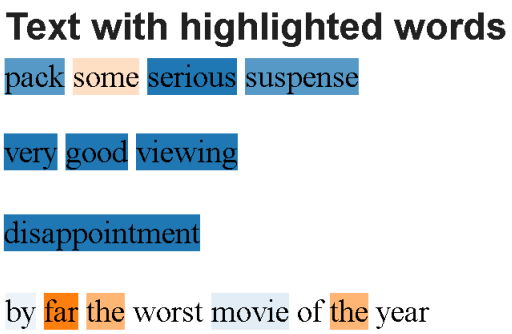

Figure 6: Result of Lime on CNN

For the most part, the results of integrated makes sense and matches with LIME except for the word 'disappointment' which LIME gives a very positive connotation and giving the word 'worst' a neutral connotation

5.2.2 BERT

.

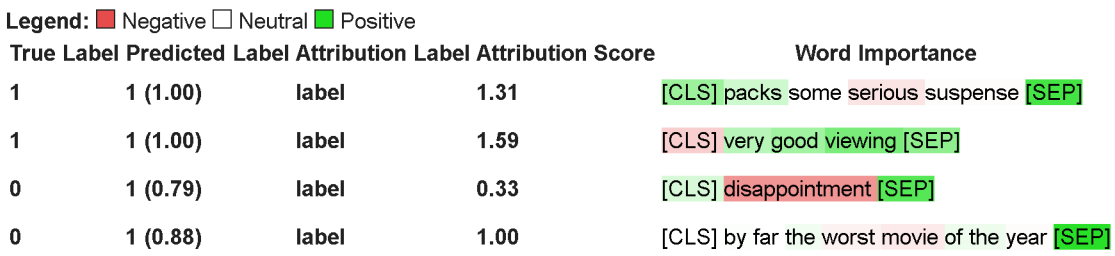

Figure 7: Result of integrated gradients on BERT

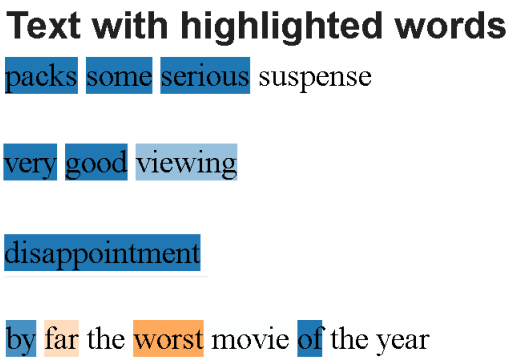

Figure 8: Result of Lime on BERT

Once again, the results of integrated gradients with BERT makes sense and matches with LIME except for, once again, the word 'disappointment' which LIME gives a very positive connotation

6 Conclusion and Future Work

From the results, it can be concluded that integrated gradients give more reliable and consistent interpretations when compared to LIME. When comparing the transformer model, BERT, to the baseline CNN model, BERT gives similar results bit it is less 'confident' and a bit inconsistent when interpreting a sentence.

More conclusive results can be provided if the models are trained on more, larger datasets. At the moment, integrated gradients doesnt handle multiclass sentiments but it would be interesting to look at how the models would interpret more than two sentiments. Finally, more transformer model can be added to the comparison and see how they are interpreting the results.