International Journal of Computer Trends and Technology
ISSN: 2231 – 2803 / https://doi.org/10.14445/22312803/IJCTT-V70I7P103

Volume 70 Issue 7, 11-15, July 2022

Original Article

Explainable AI for NLP: Decoding Black Box

Yogendra Sisodia

[1] *Director Machine Learning, Conga, Maharashtra, India.*

Received: 18 May 2022 Revised: 19 July 2022 Accepted: 23 July 2022 Published: 31 July 2022

Abstract *- Recent advancements in machine learning have sparked greater interest in previously understudied topics. As machine learning improves, experts are being pushed to understand and trace how algorithms get their results, how models think, and why the end outcome. It is also difficult to communicate the outcome to end customers and internal stakeholders such as sales and customer service without explaining the outcomes in simple language, especially using visualization. In specialized domains like law and medicine, it becomes vital to understand the machine learning output.*

Keywords *- Artificial Intelligence, Natural Language Processing, Explainable AI, Deep Neural Network, Transformers.*

1. Introduction

The size of the NLP deep learning models is increasing daily, with billions of parameters [1]. Base BERT, for example, has 110 million parameters [1]. Explaining them is hard, and they act as a complete black box, so they come at the expense of models that become less interpretable. A decade ago, techniques like Decision Tree were white box in nature. The Explainable AI (XAI) methods aim to make them end-user-friendly (domain experts), fair, and accountable. Also, further feedback is required to train them, so XAI becomes more important.

This paper discusses all the relevant and latest explainable AI methods for NLP as a mini-survey. The author implemented a wrapper called NlpExplainer to understand transformer-based classifiers in a few lines. The idea is to use it for more tasks, like answering questions and summarizing, and to make it possible to use complex libraries with less boiler code by providing simple wrappers around them [2].

2. Categorization of XAI

There are primarily two categories. The first determines whether the explanation is for a single prediction (local) or the entire prediction process of the model (global). The second distinguishes between explanations that emerge directly from the prediction process and those that require additional processing (post hoc).

2.1. Local vs. Global

A local explanation provides information or justification for the model's prediction of particular text input for classification tasks.

A global explanation provides a comparable justification by revealing how the predictive process of the model perates, independent of any specific input.

2.2. Self-Explanatory versus Ad-Hoc

Whether the explanation is local or global, explanations differ in terms of whether they emerge during the prediction process or whether their generation requires post-processing after the model has made a prediction. A strategy that is self-explanatory, also known as directly interpretable.

3. Techniques of XAI

3.1. Feature importance

The primary objective is to derive an explanation by analyzing the importance scores of various predictive features used to generate the final prediction.

3.2. Surrogate model

Model predictions are explained by learning as a proxy for a second, typically more explicable model. LIME is a well-known example [3].

3.3. Example or Similarity driven

These approaches explain the prediction of an input instance by identifying and presenting other semantically similar instances, typically from available labeled data.

3.4. Provenance-based

Explanations are provided by illustrating some or all of the prediction derivation process, which is an effective and intuitive technique for explaining ability when the final prediction results from a sequence of reasoning steps.

3.5. Declarative induction

Human-readable representations such as rules, trees, and programmes are induced as explanations.

4. Results and Discussion

4.1. Shap Values

SHAP values are a standardized indicator of feature importance for any classification task. These are the Shapley values of the original model's conditional expectation function [4].

The increasing trade-off between the precision and interpretability of deep learning-based model predictions has resulted in the development of techniques that assist machine learning engineers and end-users in interpreting predictions, thus deciphering the Black Box. The SHAP framework illustrates the class of additive feature importance methods (which includes a set of previous methods). It demonstrates a unique solution within this class that adheres to desirable properties. As shown by the way SHAP connects different pieces of literature, common model interpretation principles can be used to guide the development of new methods.

Shap values attribute the expected change in the model's prediction to each feature when that feature is considered. They explain how to get from the predicted base value E[f(z)] to the current output f(x) if no features were known.

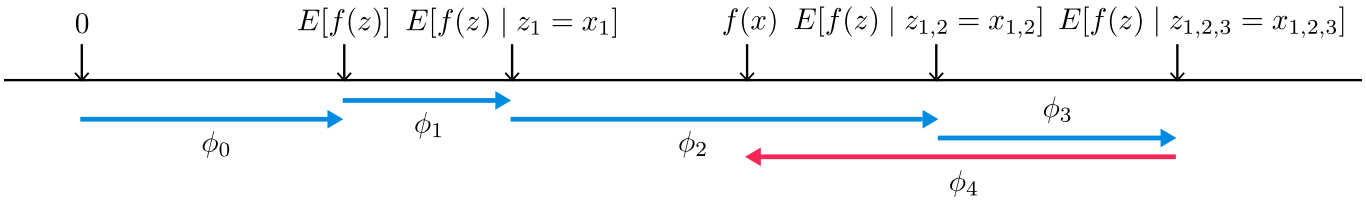

Fig. 1 SHAP (SHapley Additive exPlanation) [4]

4.2. Transformers Architecture

The Transformer [6] has swiftly surpassed alternative neural architectures to become the dominant structure for natural language processing. Models such as CNN and RNN were dominant choices before the advent of Transformers.

Transformers contain Encoder and Decoders stacks. The encoder on the left side of the Transformer architecture (shown in Figure 2) maps an input sequence to a sequence of continuous representations, which are then sent to a decoder for further processing.

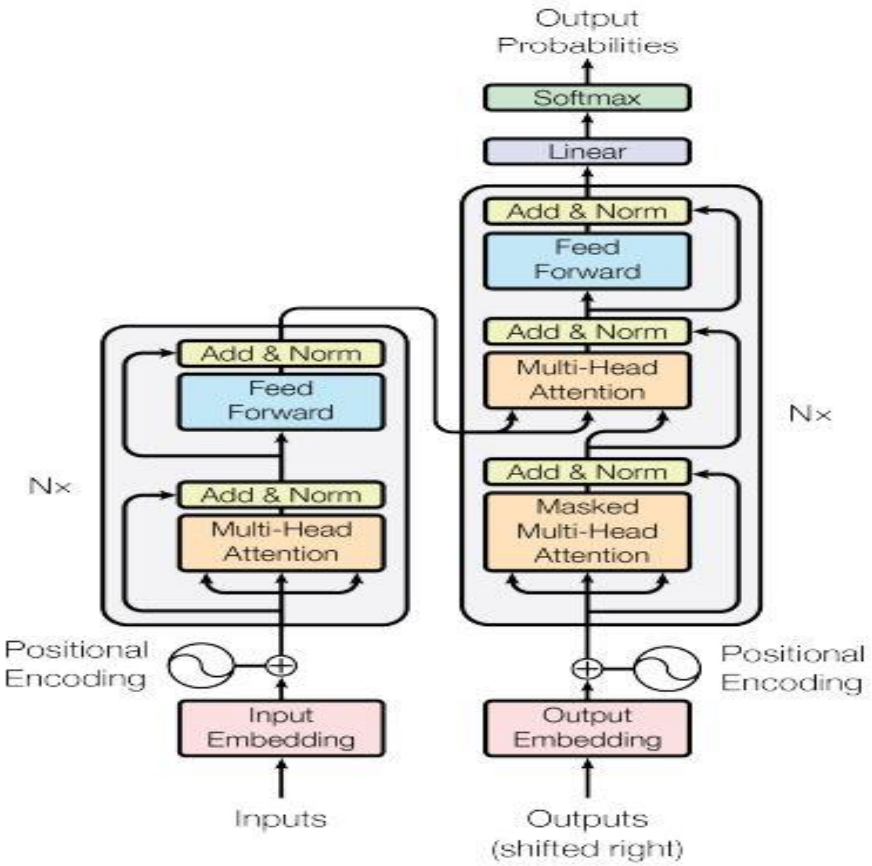

Fig. 2 Transformers Architecture [7]

On the right-hand side of the architecture, the decoder receives the encoder's output along with the decoder's output from the previous time step to generate an output sequence.

One of the key examples of transformers is BERT [7], which can be fine-tuned to specific domains and trained on several different tasks.

The author introduced NlpExplainerShapTransformers class which is a Wrapper abstracting Shap and Transformers for the classification task. The idea is to reduce no lines of code and only democratize the different algos for pragmatic usage [8].

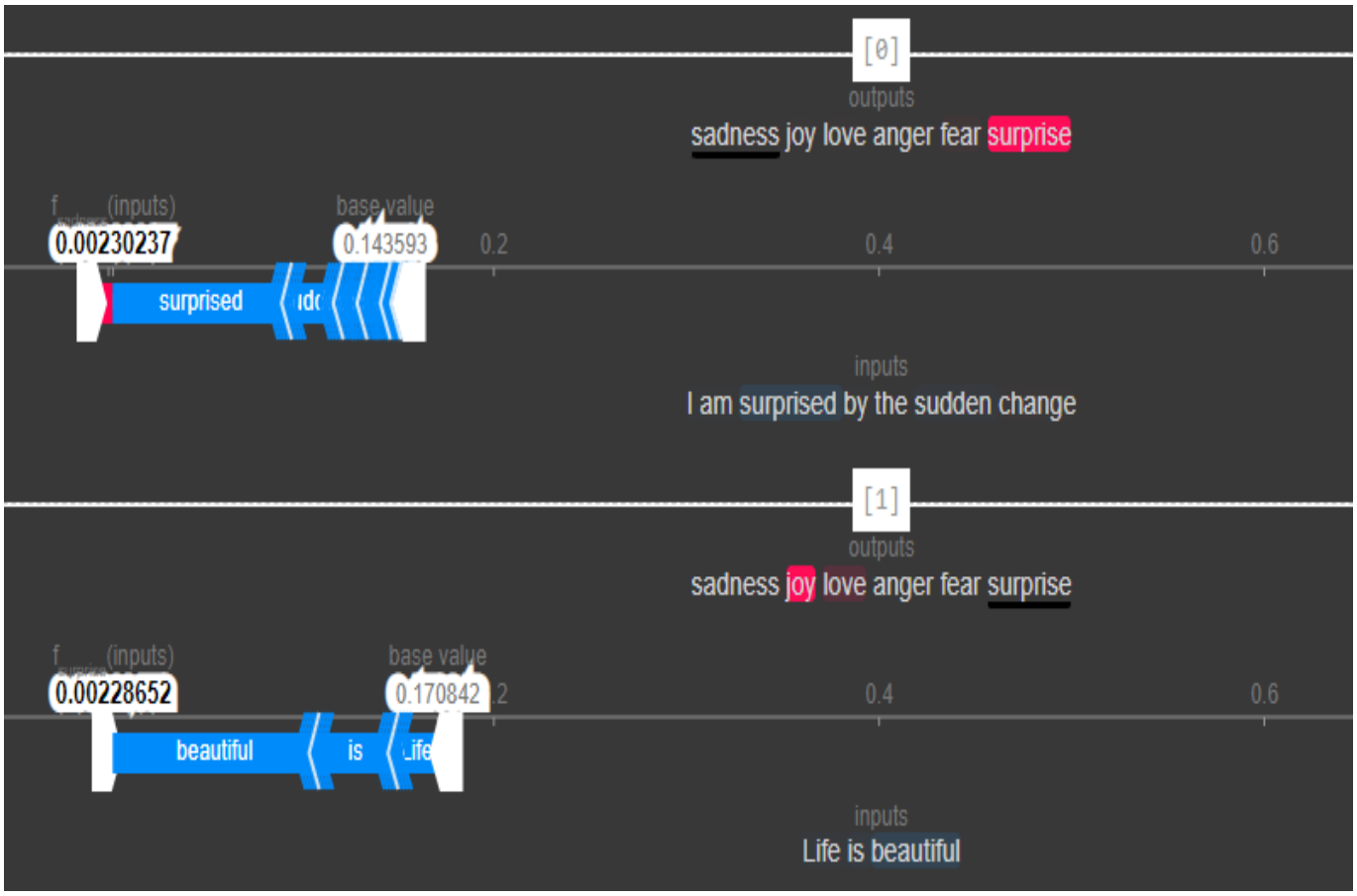

Fig. 3 For two sentences, we want to see the impact of different words like 'surprised' and 'beautiful' on classes like 'surprise' and 'joy.'

This diagram demonstrates using the library for a multiclass text classification scenario. After computing the SHAP values for a set of target sentences, we visualize the feature attributions for each class used for the classification. We use a BERT model fine-tuned on an emotion dataset to classify a sentence into six classes: joy, sadness, anger, fear, love, and surprise. This is the standard example of the SHAP library, but the author has tested it with multiple datasets, such as IMDB sentiment analysis, and it can be used for any custom class.

We compute Shap values and explainer and visualize the impact on output classes. The model outputs the base value when the entire input text is masked, while the output class is the model's output for the full original input. The SHAP values explain in an additive way how the impact of unmasking each word changes the model output from the base value (where the entire input is masked) to the final prediction value.

Similarly, we can plot the top words impacting a specific class.

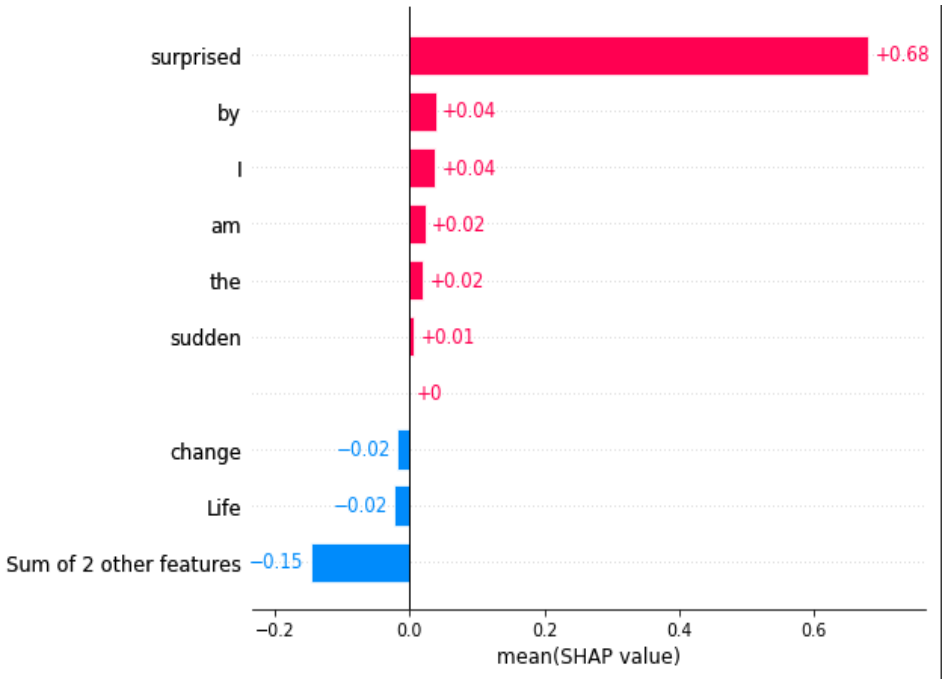

Fig. 4 Top words impacting a specific class. Here 'surprised' word brings the highest contribution to the 'surprise class, followed by 'by' and 'I.' We can create all these diagrams using SHAP libraries directly, but NLPExplaier hides all complexity of the boilerplate. We can get an explainer, shap values, and predictions in a few lines.

The below code snippet shows an example by which we can get the following in a few lines of codes

- Labels of Classifiers like Joy/Surprise
- Shap Values for each Token and Class
- Predictions for each class

5. Conclusion

Much effort is going into explaining deep neural network-based models like Transformers. Shap Values is one of them. The author talked about major categories, categorization, and techniques. Also, the author introduced a library called NlpExplainer for the same. Currently, it works only for classification and shap. Future work involves incorporating more tasks like QnA and summarization and implementing more techniques like LIME apart from Shap.

References

[1] Tom B. Brown Et Al, "Language Models Are Few-Shot Learners," in: Arxiv Preprint Arxiv:2005.14165v4 , 2020.
[2] Shap Library [Online]. Available Https://Github.Com/Slundberg/Shap/
[3] Lime Library [Online]. Available Https://Github.Com/Marcotcr/Lime
[4] Scott Lundberg, Su-in Lee, "A Unified Approach To Interpreting Model Predictions," in: Arxiv Preprint Arxiv:1705.07874v2, 2017.
[5] Avanti Shrikumar Et Al, "Not Just a Black Box: Learning Important Features Through Propagating Activation Differences," in: Arxiv Preprint Arxiv:1605.01713, 2016.
[6] Ashish Vaswani Et Al, "Attention Is All You Need," in: Arxiv Preprint Arxiv:1706.03762v5 , 2017.
[7] Jacob Devlin Et Al, "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding," in: Arxiv Preprint Arxiv:1810.04805v2, 2018.
[8] Nlpexplainer Library [Online]. Available Https://Github.Com/Scholarly360/Nlpexplainer
[9] Eneko Agirre, Llu'is M'Arquez, and Richard Wicentowski, "Proceedings of the Fourth International Workshop on Semantic Evaluations" (Semeval), 2007.
[10] Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. "Clozedriven Pretraining of Self-Attention Networks," Arxiv Preprint Arxiv:1903.07785, 2019.
[11] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. "A Large Annotated Corpus for Learning Natural Language Inference," 2015.
[12] William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. "KERMIT: Generative Insertion-Based Modeling for Sequences," Arxiv Preprint Arxiv:1906.01604 , 2019.
[13] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S.Weld, Luke Zettlemoyer, and Omer Levy, "Spanbert: Improving Pre-Training By Representing and Predicting Spans," Arxiv Preprint Arxiv:1907.10529 , 2019.
[14] Diederik Kingma and Jimmy Ba Adam, "A Method for Stochastic Optimization", *In International Conference on Learning Representations (ICLR)*, 2015.

[15] Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu,Yordan Yordanov, and Thomas Lukasiewicz. "A Surprisingly Robust Trick for Winograd Schema Challenge," Arxiv Preprint Arxiv:1905.06290, 2019.
[16] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. "Race: Large-Scale Reading Comprehension Dataset From Examinations," Arxiv Preprint Arxiv:1704.04683 , 2017.
[17] Guillaume Lample and Alexis Conneau," Crosslingual Language Model Pretraining," Arxiv Preprint Arxiv:1901.07291, 2019.
[18] Hector J Levesque, Ernest Davis, and Leora Morgenstern, "The Winograd Schema Challenge," *in AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, 2011.
[19] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao, "Improving Multi-Task Deep Neural Networks Via Knowledge Distillation for Natural Language Understanding," Arxiv Preprint Arxiv:1904. 9482 , 2019.