Interpretation of Neural Models using LRP and Integrated Gradient Approach

Rricha Jalota
Matriculation Nr. 7010592
rrja00001@stud.uni-saarland.de

Abstract

With the increasing popularity of neural networks for solving complex tasks, understanding and explaining the behaviour and predictions of these models has become a crucial topic for research. To this end, several model interpretability methods and frameworks have been introduced. In this work, a couple of such methods have been studied for the task of text classification. In particular, a gradient-based method, Integrated Gradients, and a propogation-based method, Layer-wise Relevance Propogation, have been employed to interpret a CNN classifier trained on Stanford Sentiment Treebank.

1 Introduction

As neural networks are gaining popularity for automating not just day-to-day but also critical tasks across all sectors - financial, healthcare, education, retail, etc., it is essential to understand what's happening under the hood. Understanding how a black-box neural network makes predictions would not just help in debugging or testing a model's performance under various constraints, but also 1) shed some light on what a model is actually learning (i.e. identify "Clever Hans" predictors), 2) lead to new insights, 3) ensure transparency and thus, foster trust amongst end-users (Samek et al., 2019).

Recently, several techniques have been introduced to explain a model's predictions. Some of these techniques are model agnostic and rely on surrogate functions to explain the predictions (Ribeiro et al., 2016; Smilkov et al., 2017) or they generate explanations by investigating a model's response to local changes (gradient or perturbation-based approaches) (Fong and Vedaldi, 2017; Sundararajan et al., 2017). On the other hand, some techniques rely on the internal structure of the model (propogation-based approaches) to explain predictions (Bach et al., 2015; Shrikumar et al., 2017). A more detailed account on all of these approaches can be found in the Explainable AI book (Samek et al., 2019).

Since majority of these techniques have been extensively studied on vision tasks, their adaptation to NLP tasks holds an interesting premise. This is because, in NLP tasks, the features are not pixels anymore (which can be easily perturbed) but rather word representations that hold meaning. The objective of this work, therefore, is to explain the predictions of a text classifier. In particular, a convolutional neural network (Kim, 2014a) trained on the task of Sentiment Analysis is explained using a gradient-based approach - Integrated Gradients (Sundararajan et al., 2017) and a propogation-based approach - Layer-wise Relevance Propogation (Binder et al., 2016). Furthermore, the effect of different training epochs on the relevance of the words w.r.t. a prediction is also investigated via heatmap visualizations.

The rest of the paper is structured as follows: in section 2, the background concepts and related work have been discussed. Section 3 addresses the text-classification and interpretibilty approaches used. In section 4, results from the experiments are presented and discussed. Finally, section 5 concludes the work and lists ideas for future work. The related code can be found at https://github.com/rrichajalota/Interpreting-CNN-BiLSTM.

2 Background and Related Work

2.1 Text Classification and Sentiment Analysis

Sentiment Analysis is the process of analyzing a piece of text to determine whetrher the writer's opinion towards a topic is positive, negative or neutral. It is an important area of NLP Research because this automation helps in the advancement of

several economic sectors. Sentiment analysis is inherently a text classification problem and has been widely studied in the recent years. To represent text, classification approaches either employ: 1) explicit, hand-engineered features (Sriram et al., 2010) to enrich the text representation, or 2) neural networks to model implicit dependencies within the text (Wang et al., 2017). While it is easier to interpret statistical models like decision trees, model interpretation becomes increasingly difficult as the complexity (non-linearity) increases with neural networks. Though attention mechanism (Vaswani et al., 2017) and pre-trained language models like BERT (Devlin et al., 2018) are becoming prevalent for the task of text classification, simpler neural network models like Convolutional Neural Networks (Kim, 2014b) have also shown decent performance over various text classification tasks. Therefore, in this study, the focus is on this classification approach for the task of Sentiment Analysis.

2.2 Intrepretability Algorithms

Interpretability methods that have been studied in the litertaure so far, have been described in terms of either capturing the decision-making process of a model i.e. how a model views an input and transforms it to produce an output. This is referred to as global interpretability. Or, they focus on explaining the model behavior for a single instance only, referred as local interpretability (Doshi-Velez and Kim, 2017). While global interpretability is hard to achieve (due to increasing number of model parameters), several methods for local interpretability have been introduced (Ribeiro et al., 2016; Smilkov et al., 2017; Fong and Vedaldi, 2017; Sundararajan et al., 2017; Bach et al., 2015; Shrikumar et al., 2017). In this work, two different classes of local interpretability methods have been studied.

1. **Attribution-based methods** - In the context of text classification, given a set of words in a sentence, it is desirable to know which of the input features (words) have a greater impact on the predicted output, along with the polarity of their influence. Attribution methods aim at producing explanations by assigning a scalar attribution value/relevance score, to each input word for a given input sentence (Samek et al., 2019). *Gradient*Input* (Shrikumar et al., 2016), *Integrated Gradients*, *ϵ-LRP* are some of the popular attribution-based methods. To compute the relevance score, they all require partial derivatives of the output w.r.t. the input. In this work, *ϵ-LRP* (Bach et al., 2015), *Integrated Gradients* (Sundararajan et al., 2017) have been studied, which will be elaborated in section 3.

2. **Propogation-based methods** - These methods operate by propogating a prediction backwards in the neural network layer by layer, using a set of designed propogation rules. Thus, to compute attribution maps, in addition to a forward pass, they require a backward pass through the network. Recent variants of LRP - $\alpha\beta$-LRP (Montavon et al., 2018) and DeepLift (Shrikumar et al., 2017) fall under this category and cannot be moulded to fit in the definition of gradient-based methods. (Samek et al., 2019) In this work, $\alpha\beta$-LRP has been investigated and will be further elaborated in section 3.

Both Integrated gradients and layer-wise relevance propogations (LRP) are salience methods. A salience method describes the marginal effect of a feature to the output with respect to the same input where such feature has been removed (Samek et al., 2019). In order to depict the absence of a feature, a baseline value is used, which could be the index of `<pad>` tokens for text classifiers.

3 Approach

3.1 Training the Sentiment Classifier

- **TextCNN** - A simple CNN model is trained with a single convolutional layer on top of pre-trained word vectors [1]. The architecture used to train the classifier is shown in Figure 1 and was previously experimented by (Arras et al., 2017). Word embeddings of every two words are convolved and passed to a max pooling layer, followed by a softmax layer. The hyperparameters are set as per the `iNNvestigate` (Alber et al., 2019) Sentiment Analysis tutorial [2]. Since this is a multiclass-classification problem, categorical-crossentropy has been used as the loss function. The classifier has been implemented in

[1] The pre-trained word vectors were taken from https://github.com/ArrasL/LRP_for_LSTM/raw/master/model/

[2] https://github.com/albermax/innvestigate/blob/master/examples/notebooks/sentiment_analysis.ipynb

both PyTorch and Keras, and shows a similar performance on the test dataset. However, only the classifier trained on Keras is discussed in this paper. The classifier trained on Pytorch can be found in the code repository[3].

3.2 Interpreting the Classifiers

The three techniques that have been used to interpret the classifiers have been mentioned below.

1. **Integrated Gradients** compute a contribution score (or attribution value) for each feature by taking the integral of the gradients along a straight path from a baseline instance $\hat{x}$ to the input instance x. For classification models, the gradient is generally taken w.r.t. the output corresponding to the true class or to the class predicted by the model. To formalize, consider an input instance x, a baseline instance $\hat{x}$ and a model $M : X \rightarrow Y$ which acts on the feature space X and produces an output y in the output space Y. A function F for multi-class classification, is then defined as $F(x) = M_k(x)$; where the model output is a vector of probabilities, with the index k denoting the k-th element of $M(x)$. The attributions $A_i(x, \hat{x})$ for each feature x_i with respect to the corresponding feature $\hat{x}_i$ in the baseline are calculated as

$$A_i(x, \hat{x}) = (x_i - \hat{x}_i) \int_0^1 \frac{\partial F(\hat{x} + \alpha(x_i - \hat{x}_i))}{\partial x_i} d\alpha \quad (1)$$

here the integral is taken along a straight path from the baseline $\hat{x}$ to the instance x parameterized by the parameter α[4].

2. **ϵ-LRP** - Layer-wise relevance propagation is a backpropagation technique for interpreting deep neural networks. It analyzes how each word of a sentence contributes to the classification function through each layer. Consider a quantity r_i^l, which refers to the relevance of unit i of layer l. The LRP algorithm starts at the output layer L. It assigns the relevance of the target neuron c equal to the activation of the neuron itself, and the relevance of all other neurons to 0. Then, it traces backwards layer by layer, redistributing the prediction score S_i until the input layer is reached. Figure 2 shows the formula that is used to calculate the attributions at each layer. The value in the numerator $z_{ij} = r_i^l w_{ij}^{(l,l+1)}$ which is the weighted activation of a neuron i onto neuron j in the next layer. A small value, called epsilon, is added in the denominator to avoid numerical instabilities (Samek et al., 2019). ϵ-LRP can be reformulated as the feature-wise product of the input and the ratio between the output and the input at each non-linearity (modified gradient). Hence, this method is considered as a gradient-based attribution method.

3. **$\alpha\beta$-LRP** treats the activating and inhibiting neurons separately by setting the parameters - alpha and beta (Lapuschkin et al., 2016). The formula is shown in Figure 3.

4 Experiments

To carry out all the experiments, the Stanford Sentiment Treebank dataset (Socher et al., 2013) has been used with the given train/dev/test splits and fine-grained labels - *very positive, positive, neutral, negative, very negative*. The results of the classifiers trained on 5, 10 and 20 epochs are shown in Table 1.

Table 1: Evaluation results of the trained classifier with dropout 0.25

Classifier	num_epochs	test_loss	test_accuracy
CNN	5	1.28	0.431
CNN	10	1.27	0.438
CNN	20	1.32	0.436

After several trials of experiments with the Pytorch library `Captum` (Kokhlikyan et al., 2020), I decided to switch to `iNNvestigate` (Alber et al., 2019) for interpreting the trained CNN classifier. The reasons are two-fold: 1) I found the documentation for `iNNvestigate` more complete and coherent with its examples, 2) it has support for more analysis methods than Captum (LRP is not currently implemented in the Captum framework)[5]. Figures 4, 5 and 6 show heatmaps generated for three sets of experiments - classifiers trained for 5,

[3] `ABiLSTMmodelisalsopresentintherepo. IthasbeentrainedinPytorchandinterpretedusingCaptumlibrary sintegratedgradientsmethod.`

[4] Soure: `https://docs.seldon.io/projects/alibi/en/stable/methods/IntegratedGradients.html`

[5] My experiments with the Captum framework in Pytorch can still be found in the code repo

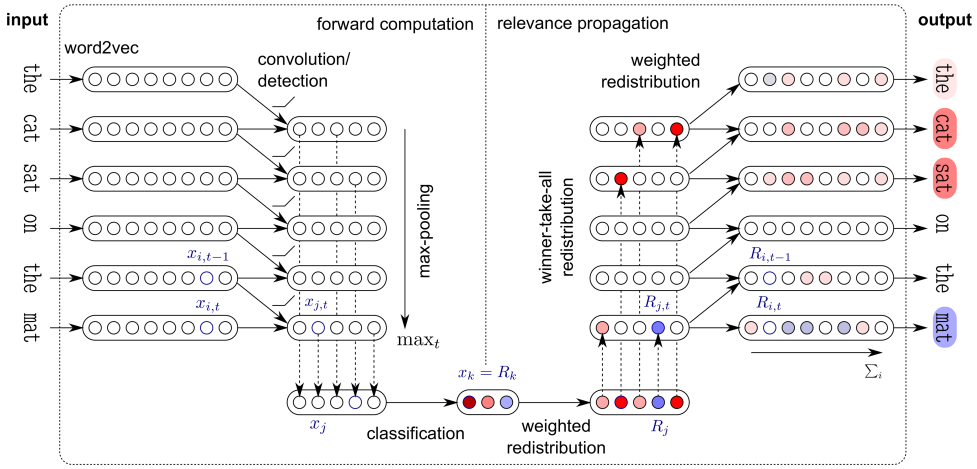

Figure 1: Interpreting a CNN text-classifer. Figure taken from (Arras et al., 2017).

$$r_i^{(L)} = \begin{cases} S_i(x) & \text{if unit } i \text{ is the target unit of interest} \\ 0 & \text{otherwise} \end{cases}$$

weighted activation of neuron i of layer l onto neuron j of layer l+1

$$r_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_{i'} z_{i'j} + b_j + \epsilon \cdot sign(\sum_{i'} z_{i'j} + b_j)} r_j^{(l+1)}$$

bias of unit j

Figure 2: Formula for ϵ-LRP. Source (Samek et al., 2019)

$$R_i = \sum_j \left(\alpha \cdot \frac{(x_i w_{ij})^+}{\sum_i (x_i w_{ij})^+} - \beta \cdot \frac{(x_i w_{ij})^-}{\sum_i (x_i w_{ij})^-} \right) R_j,$$

Figure 3: $\alpha\beta$-LRP rule redistributes relevance from layer $l+1$ to layer l. $()^+$ and $()^-$ show the positive and negative parts. Source (Samek et al., 2016)

10 and 20 epochs with adam optimizer (learning rate=0.001, batch_size=256). The words that are highlighted in strong red have high positive relevance score and the ones that are shaded in blue have negative score with respect to the prediction. For all the methods, gradient is computed with respect to the predicted class.

5 Discussion

From the Table 1, it can be inferred that, with a learning rate of 0.001, the model starts overfitting as the number of epochs are raised from 5 to 20. This was also prominent from the training and validation loss (not included in the report but can be seen in the jupyter notebook). So we now have an interesting scenario for model debugging and investigating whether the input features contribute differently to the model training as the number of epochs are increased. All of the three trained models were therefore interpreted using the three algorithms discussed in Section 3.2.

From the heatmap figures 4, 5 and 6, it can be seen that while the heatmaps for epoch 5 and 10 are almost identical, there is a noticeable difference in the heatmaps of epochs 10 and 20. In particular, the heatmaps for ids= 30, 23 in figures 5 and 6 show difference in the intensity of coloration. As we know, that the model has overfitted with 20 epochs, these attribute maps indicate how the contribution of input words have changed towards the final prediction. Even though the predicted outputs for both of these review ids (23 and 30) haven't change across the epochs, the intensity of colouration depicts the contribution of input words. It can be seen that, out of the three model interpretability algorithms, integrated gradients shows a significant difference in the attribution of the input words (not just the intensity but also the color changes). For review-id 23, ϵ-LRP also shows a variation in the color intensity. While these few samples are not representative of the entire test set, this small experiment does indicate that the algorithms for model interpretability and especially, integrated gradients method can be used for model debugging and gain-

Figure 4: Analysis of the predictions from a CNN model trained on 5 epochs.Words that have a positive score to the prediction are be shaded in red, while negative-contribution or zero-contribution words are then highlighted in blue, and white, respectively.

ing insights into how a model is actually learning.

6 Conclusion and Future Work

In this work, a sentiment classifier trained on a simple convolutional neural network, for different number of epochs, has been interpreted using three explainability techniques - Integrated Gradients, ϵ-LRP and $\alpha\beta$-LRP. The aim of this study was to investigate how different interpretability algorithms can be used to gain insights into the model behaviour and performance. In particular, the idea was to check, for the given case-study, which interpretability technique gives better insights into the model behaviour. The experiments and heatmap visualizations indicate Integrated Gradients method to be more efficient than the other two layer-wise propogation approaches. However, this premise cannot be generalized based on the toy experiments performed in this study i.e. on just one classifier with controlled settings (hyper-parameters). More experiments need to be performed to better understand the workings of all the three interpretability approaches. Due to time constraints, experiments could not be performed on BiLSTM with the `iNNvestigate` framework. However, this accounts for future work. I plan to train a BiLSTM model in keras and then apply these methods to check whether they perform similarly on the BiLSTM model. For BiLSTM, layer-wise propogation algorithms must show a different result as these techniues are not model-agnostic. Furthermore, it would also be interesting to see how the change in word embeddings (fastText or BERT), changes the relevance of the input features w.r.t. model predictions.

References

Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. 2019. innvestigate neural networks! *Journal of Machine Learning Research*, 20(93):1–8.

Leila Arras, Grégoire Montavon, Klaus-Robert Müller,

Figure 5: Analysis of the predictions from a CNN model trained on 10 epochs. Words that have a positive score to the prediction are be shaded in red, while negative-contribution or zero-contribution words are then highlighted in blue, and white, respectively.

and Wojciech Samek. 2017. Explaining Recurrent Neural Network Predictions in Sentiment Analysis. In *Proceedings of the EMNLP 2017 Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 159–168. Association for Computational Linguistics.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.

Alexander Binder, Sebastian Bach, Gregoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016. Layer-wise relevance propagation for deep neural network architectures. In *Information science and applications (ICISA) 2016*, pages 913–922. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437.

Yoon Kim. 2014a. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Yoon Kim. 2014b. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch.

Sebastian Lapuschkin, Alexander Binder, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016. The lrp toolbox for artificial neural networks. *The Journal of Machine Learning Research*, 17(1):3938–3942.

Figure 6: Analysis of the predictions from a CNN model trained on 20 epochs. Words that have a positive score to the prediction are be shaded in red, while negative-contribution or zero-contribution words are then highlighted in blue, and white, respectively.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2018. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Wojciech Samek, Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, and Klaus-Robert Müller. 2016. Interpreting the predictions of complex ml models by layer-wise relevance propagation. *arXiv preprint arXiv:1611.08191*.

Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. 2019. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. 2016. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher Manning, Andrew Ng, and Christopher Potts. 2013. Parsing With Compositional Vector Grammars. In *EMNLP*.

Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. 2010. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining knowledge with deep convolutional neural networks for short text classification. In *IJCAI*, pages 2915–2921.