SELF EXPLAINING STRUCTURES IMPROVE NLP MODELS: REPRODUCIBILITY STUDY

Rathi Kashi, Dhyay Bhatt & Brian Guo
Duke University
{rsk40,dpb30,bg181}@duke.edu

ABSTRACT

Existing deep learning models are not self explainable and require an additional probing model. The interpretability of these models rely on word level saliency scores and do not consider higher level text units (phrases or paragraphs). The paper proposed a self-explaining structure that used an interpretation layer to gather information about text spans of various levels and attach an importance score to them, this weighted combination was passed to the softmax function to predict the output. The paper demonstrated this novel model's success in interpretability and improvement in accuracy on the SST-5 and SNLI datasets. We aim to reproduce these results in our paper.

1 INTRODUCTION

This project is a reproducibility study on the 'Self Explaining Structures improve NLP Models' paper (Sun et al., 2020). Traditional neural networks are black box models that do not provide insight into in the decision making process of a model which raises ethical concerns, hinders model behaviour and error analysis and limits its scope in research fields such as drug discovery and genomics (Zhang et al., 2021). Interpretability of models has thus become a desired property with parliaments, such as the EU, adopting frameworks on ethical rules for AI (201, 2019). Existing Interpretability models have two main issues: They are not self-explainable and need an external probing model which is not as accurate and is less efficient. Secondly, they usually only focus on learning word-level importance scores, which often ignore the complex semantic nature of phrases and encode much less information. The three conditions for a model to be a good interpretable model (as described in the paper) are that the model should be self-explaining without an additional probing model, the model should offer clear saliency scores for any text unit level, the self-explaining nature does not trade off performance. To this end, the paper introduces the self-explaining model which consists of an interpretation layer that aggregates text spans and assigns each of them weights, and these weighted combinations are passed into a softmax function in order to get a final prediction. On reproducing the models described in this paper, we obtained a score of 56.47 on SST-5 and 91.04 on the SNLI dataset.

2 RELATED WORK

There have been several approaches that try to understand models in NLP, the first papers in the field, extracted pieces of input from the input text, that they defined as rationales. These pieces of input text were given as justification for the model's predictions (Lei et al., 2016) (Cheang et al., 2020). Some approaches were more visual in trying to understand structures such as LSTMs (Karpathy et al., 2015) or CNNs (Selvaraju et al., 2017). These papers used varying techniques in order to determine what the saliency scores should be. The most common method that has been used is called probing; this technique uses another model, such as a linear classifier (Alvarez Melis & Jaakkola, 2018) or a data structure like a knowledge graph(Anelli et al., 2022) in order to interpret the main model and extract information from what it is doing. However, only partial information can be extracted using this technique. Most models use 'saliency methods' that look for salient features responsible for the model's prediction. These give us insight into the word/s or phrases that are responsible for the decision. There are various techniques that can be used in order to

compute these saliency scores, one approach being the use of saliency maps to identify and extract task-specific salient sentences from documents to preserve document topics and semantics (Li et al., 2015), while another approach is handcrafting adversarial examples to find the most salient text-editing operations in a sentence (Ebrahimi et al., 2017). These models and approaches are most commonly used in interpretation. The most prominent benchmark to evaluate these interpretability models with each other, is the ERASER Benchmark (DeYoung et al., 2019). This benchmark tests models against a range of datasets and tasks to see how well they fare with respect to how a human would interpret those sentences, and on how generalizable they are. The problem, however, with most of these models is that they are not self-explaining; they have a probing model, and this reduces the accuracy and performance of the model, as a second probing model has to be trained along with the main model. The paper we aim to reproduce, was inspired by Alvarez Melis & Jaakkola (2018)'s 'self explaining' by jointly training a deep learning model and a linear regression model with human interpretable features and Selvaraju et al. (2017), which computes the gradient with respect to feature maps of the high-level convolutional layer to obtain high-level saliency scores.

3 Approach

3.1 Background and Notations

Our input sequence is defined as follows: $\boldsymbol{x} = \{x_1, x_2, ..., x_N\}$ where N is the length of $\boldsymbol{x}$. $\boldsymbol{x}(i, j)$ is the text span $\{x_i, x_{i+1}....x_j\}$. Our goal is to predict label $y \in Y$ given $\boldsymbol{x}$ based on $p(y|\boldsymbol{x})$. V represents vocabulary size and word representations are stored in $\mathbf{W} \in \mathbb{R}^{V \times D}$.

We followed the model as described in the paper which used a transformer as the base model in order to perform Sentiment classification, Natural Language Inference, and Machine Translation. (The base model can be replaced by LSTMs or CNNs).

3.1.1 Model Structure

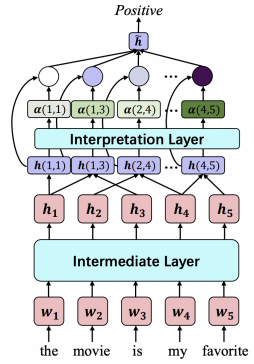

Figure 1: Model proposed by the paper

Input Layer: The first layer is the input layer which consists of a stack of word representation for each word in the input sentence (x_t is a D dimensional vector $\mathbf{W}[x_t, :]$).

Intermediate Layer: The second layer is the intermediate layer, this layer has K encoder stacks, and each of these stacks have multi head attention, layer normalization and residual connections. Each word when passed through this layer can then generate a dense embedding corresponding to that word that can be defined as h_t (where $h_t \in \mathbb{R}^{K \times D}$).

SIC Layer: The third layer is the Span Infor Collecting Layer which enables direct measurement of a text span. This layer takes in a hidden representation $h(i, j)$ which correspond to the text span $x(i, j)$. $h(i, j)$ is obtained by taking the representation of the starting index at the last intermediate layer and the ending index at the last intermediate layer and computing a function over the two. This

is described mathematically as follows:

$$h(i,j) = F(h_i^K, h_j^K) \tag{1}$$

$$F(h_i, h_j) = tanh[\mathbf{W}(h_i, h_j, h_i - h_j, h_i \bigodot h_j)] \tag{2}$$

Where F is a feed forward network. Here, $\mathbf{W} = [W_1, W_2, W_3, W_4]$ and $W_i \in \mathbb{R}^{D \times D}$ ($\bigodot$ represents pairwise dot between two vectors). The above four elements capture concatenation, element-wise difference and element-wise closeness between the two vectors. This layer iterates over all text spans and $i \in [1, N]$ and $j \in [1, N]$ and collects all $h(i,j)$s.

Interpretation Layer: The fourth layer is the Interpretation Layer, this is at the core of the model which takes the hidden spans $h(i,j)$ and assigns them weights which can be represented as $\alpha(i,j)$ these weights can be calculated by mapping $h(i,j)$ to a scalar and then normalizing the exponent of these products to $\alpha(i,j)$. This layer then outputs the weighted average by summing over the product of $h(i,j)$ and $\alpha(i,j)$ and summing this product from i to j. Mathematically, this is:

$$o(i,j) = h^T.h(i,j) \tag{3}$$

$$\alpha(i,j) = \frac{\exp(o(i,j))}{\Sigma_{i,j} \exp(o(i,j))} \tag{4}$$

$$\tilde{h} = \Sigma_{i,j} \alpha(i,j) h(i,j) \tag{5}$$

Output Layer: This is then passed into the fifth and final layer which is the output layer, this layer takes the outputs from the previous interpretation layer and passes it into the softmax function which gives the probability distributions of the spans. Thus the span with the highest probability affects the sentence the most.

$$p(y|x) = \frac{u_y^T \tilde{h}}{\Sigma_y u_y^T \tilde{h}} \text{ where } u_y \in \mathbb{R}^{D \times 1} \tag{6}$$

F, the mapping function that is used in the SIC layer is defined as above to ensure that the runtime computing the complexity of one text span is $O(N^2 D)$ Where N is the length of the input sentence and D is the length of the text-span, this makes this model fairly efficient.

3.2 Method

This model can be trained using standard cross entropy loss, the model will train the best when it is exposed to a small number of text spans. A regularization term is added to the training objective as follows:

$$L = \log p(y|x) + \lambda \Sigma_{i,j} \alpha^2(i,j) \tag{7}$$

3.3 Evaluation

The paper claims that if the model extracts rationales (text spans that have the maximum contribution to the input text) that can faithfully represent the input text, then a model trained on the input text should do well when tested on the rationales, a model trained on the rationales should do well when tested on the input text and a model trained on the rationales should do well when tested on the rationales. To this end, the paper creates datasets FullTrain, SpanTrain, FullTest and SpanTest and checks to see how the model does in combinations of the above. We do the same in our implementation.

3.4 Tasks and Datasets

We have conducted three NLP tasks using 3 datasets, we start off doing text classification using the SST-5 dataset; we then performed natural language inference using the SNLI dataset; finally attempted to perform machine translation tasks using the IWSLT2014 En $\rightarrow$ De dataset, we weren't able to implement machine translation successfully, and finally we use ROBERTa-base (Liu et al., 2019) as the backbone of the model and add the interpretation layer to a transformer (Vaswani et al., 2017).

We use three datasets to attempt to run various training tasks: The SST-5 Dataset, the SNLI Dataset, and the IWSLT2014 En $\rightarrow$ De dataset:

1. The SST-5 Dataset or the The Stanford Sentiment Treebank contains five classes (very negative, negative, neutral, positive and very positive) we perform sentiment classification both at phrase and word levels on this dataset.
2. The SNLI Dataset or the The Stanford Natural Language Inference corpus contains 570K examples of english sentence pairs written by humans in order to perform natural language inference, the labels here are entailent, contradiction and neutral.
3. The IWSLT2014 En $\rightarrow$ De dataset contains 167K english to german sequence pairs (160K Train and 7K Validation).We perform instructions in the paper and combine dev2010, tst2010, tst2011 and tst2011 in order to create the test set and create a joint BPE vocabulary of about 10k tokens. We were unfortunately unable to perform machine translation successfully using the IWSLT2014 En $\rightarrow$ De dataset but we will talk about our approach and what we think went wrong in the next few sections.

3.5 Extensions

We tried to replicate the model as described in the paper. The paper does not specify the hyperparameters that were used in training and testing the models, we experimented to find the best results. We were not able to successfully perform machine translation using the IWSLT2014 En $\rightarrow$ De dataset but we were able to perform the other tasks like text classification and natural language inference and we were able to get a deeper understanding of what part of the sentence influenced the result the most. (We also played around with the random general seed values in order to test if the model was truly generalizable and if the model did hold up when various variables changed). Additionally, we looked at the influence of λ on the average length of text spans and the test accuracy.

4 Experiments

4.1 Model and Training Details

The model we used is identical to the one in the paper, the paper defined the exact architecture which was the interpretation layer built into a regular transformer and we also use ROBERT-a as the backbone like in the paper. The paper did not mention hyperparameters such as the learning rate, number of epochs, batch size, and the lambda values. We trained the models on the SST5 and SNLI datasets to determine what hyperparameters worked the best. For SST5 the best prediction we got was 0.5647 using a learning rate of $2\mathrm{e}{-5}$, a batch size of 12, and 30 epochs. However using the following hyperparameters we were able to get an accuracy of 0.5643, a learning rate of $1.5\mathrm{e}{-5}$, a batch size of 10, and 20 epochs - this is almost equivalent to the best results we got but it was much quicker to run as it has 15 less epochs. For SNLI, a learning rate of $1.5\mathrm{e}{-5}$, a batch size of 12, and 10 epochs got us the best accuracy of 0.9104. We could not comprehensively test out the lambda values due to resource constraints. We used an AdamW optimizer with the linear schedule with warmup scheduler from the transformers library.

4.2 Results

These results displayed were obtained after we tuned our hyperparameters to get the best possible scores, here we compare the results we got from out SST-5 dataset and SNLI Dataset models that perform text classification and natural language inference respectively. We have benchmarked these results against scores of BERT-base (Cheang et al., 2020), RoBERTa-base+AvgSelf-Explaining, the RoBERTa-base+Self-Explaining that the authors of the paper got (Sun et al., 2020). As observed in the table for SST-5 our score is 56.47 which is higher than BERT-base (54.9) and RoBERTa-base+AvgSelf-Explaining (56.2). For the SNLI Dataset our model's accuracy at 91.04 is higher than than BERT-base (89.2) and RoBERTa-base+AvgSelf-Explaining (90.8) and is fairly close to the authors accuracy which is 91.7.

	SST-5	SNLI
BERT-base	54.9	89.2
RoBERTa-base+AvgSelf-Explaining	56.2	90.8
Paper: RoBERTa-base+Self-Explaining	57.8	91.7
Ours: RoBERTa-base+Self-Explaining replication	56.47	91.04

Table 1: Accuracy scores for various models for the SST5 and SNLI Dataset

	F-S	S-F	S-S
SST Author	38.4	54.9	42.9
SNLI Author	74.5	88.5	79.2
SST Our Team	35.8	51.3	37.69
SNLI Our Team	73.8	88.2	78.9

Table 2: Performances of different models where "F" refers to Full and "S" refers to Span.

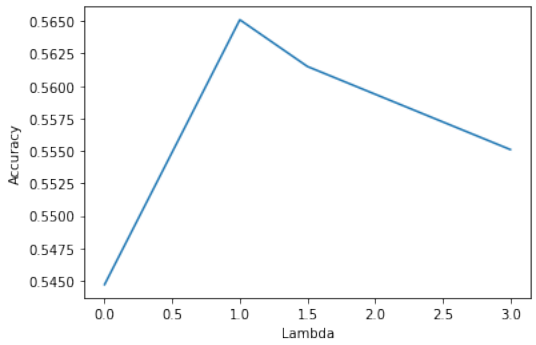

Figure 2: Lambda vs Test Accuracy on the SST-5 dataset

	Accuracy	Avg span length
$\lambda = 0$	0.5447	12
$\lambda = 1$	0.5647	8
$\lambda = 1.5$	0.5615	11
$\lambda = 3$	0.5551	10

Table 3: Performances of models with varying lambda and the average span length selected

5 Analysis

For the SST5 Dataset our model got a better score than the BERT-base and RoBERTa-base+AvgSelf-Explaining models, we were however unable to match the score that the authors were able to achieve, our score was about 1.3% less than the authors possible reasons why this may be is we had limited resources both computationally and time which is why we could not run these results on very large epochs. We also used a completely random seed which the authors make no mention of in the paper. For the SNLI Dataset our accuracy also beat the BERT-base and RoBERTa-base+AvgSelf-Explaining models and we were much closer to the authors accuracy as you can see in the table, this is a good indication of our model working fairly well.

Let us now analyse some of the outputs that the model produces, this is the interpretation part of the model where it predicts what section of the sentence produces an output that has the highest impact on the prediction in the SST model. The labels were as follows:

```
0: Very Negative
1: Negative
2: Neutral
3: Positive
4: Very Positive
```

Our model predicts the top k probabilities - it will output the top k number of spans with the highest probability of impacting the output ($k = 5$ in our implementation). Let us first look at a few good examples where the bold section is the span that is supposed to influence the prediction the most:

1. Output 1 $\rightarrow$ It might as **well have been Problem** Child IV.
2. Output 4 $\rightarrow$ The Movie **will reach far beyond its core demographic**
3. Output 0 $\rightarrow$ It's **just incredibly dull**
4. Output 2 $\rightarrow$ Both awful **and appealing**

However, not all of the predictions it makes are good, here are some examples where the span seemingly has no bearing on the label predicted:

1. Output 2 $\rightarrow$ **A** land rover is being driven across a river. A vehicle is crossing a river.
2. Output 2 $\rightarrow$ **Three** firefighter come out of subway station. Three firefighters coming up from a subway station.
3. Output 1 $\rightarrow$ **A young boy wearing a red shirt and jeans** stands in the middle of a field and throws a toy plane in the air. A young boy is playing in the field because his mother kicked him out of the house.

It is also interesting to note that sometimes the top k spans make different or even contradictory predictions about the spans. Let us take these two examples, the output for them is 1 (negative)

1. **A** statue at a museum that no seems to be looking at. The statue is offensive and people are mad that it is on display.
2. A statue at a museum that no seems to be looking at. The statue **is offensive and** people are mad that it is on display

Only one of them is correct and they both predict completely different spans, this is why it is useful to calculate more than just the top span.

If we consider examples where the model incorrectly classifies the sentiment, we observe the following:

1. Perhaps no picture ever made has **more literally showed that the road to hell** is paved with good intentions. [Correct: 3, Predicted: 2]
2. Son of **the Bride may be a good half**-hour too long but comes replete with a flattering sense of mystery and quietness. [Correct: 1, Predicted: 3]
3. 4 friends, 2 couples, 2000 miles, and **all the Pabst** Blue Ribbon beer they can drink - it's the ultimate redneck road-trip. [Correct: 3, Predicted: 0]
4. It is nature against progress. [Correct: 2, Predicted: 1]

We observe that the model makes errors if there are both positive and negative words in a sentence and it chooses a subset containing only one of them (as in the first and second example), it might

choose a text span that seemingly has no impact on the sentiment (as in the third example), or it may not be able to choose a valid span as in the last example.

Some of the flaws of this model is that it only tends to predict spans that are usually toward the front of the sentence and ignores some of the more emotionally heavy spans that are toward the end that should have influenced the models decision more.

A sentence such as : **A man walks along** with a can of soda in his hand. A sad man walks along with a can of soda in his hand is the perfect example where the output is 1, it could have predicted "a sad man walks" instead.

With respect to the regularization parameter λ, in experiments with SST-5, $\lambda = 1$ gave the best result (as opposed to $\lambda = 1.5$ in the paper). Since λ played the role in sharpening α, we were curious to see if it had an affect on the span length selected by the model. Based on our experiment, we concluded that it did not have a direct impact on span length, but perhaps had an impact on the number of spans considered by the model in predicting the final output.

6 Conclusion

The model is a great first step into providing insight into how a neural network makes decisions, this is an efficient way to perform interpretation and is very useful especially in the future. The model we developed and trained in order to replicate the paper got fairly good results beating most other base and interpretation models however more hyperparameter tuning can be performed to get results equivalent to the authors and the Machine translation tasks can be worked on. Overall this is a much better way to approach interpretation especially when trying to practically implement it when efficiency is a key consideration.

Author Contributions

All of us were involved in most aspects of the project, with some taking the lead in certain areas: Dhyay worked on the parameter tuning, testing and final presentation and report. Brian worked on creating the model, datasets and training. Rathi worked on training the model, dataset preprocessing, experiments for the results and the report.

References

Eu guidelines on ethics in artificial intelligence : Context and implementation. 2019.

David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018.

Vito Walter Anelli, Giovanni Maria Biancofiore, Alessandro De Bellis, Tommaso Di Noia, and Eugenio Di Sciascio. Interpretability of bert latent space through knowledge graphs. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3806–3810, 2022.

Brian Cheang, Bailey Wei, David Kogan, Howey Qiu, and Masud Ahmed. Language representation models for fine-grained sentiment classification. *arXiv preprint arXiv:2005.13619*, 2020.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*, 2017.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Zijun Sun, Chun Fan, Qinghong Han, Xiaofei Sun, Yuxian Meng, Fei Wu, and Jiwei Li. Self-explaining structures improve nlp models. *arXiv preprint arXiv:2012.01786*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yu Zhang, Peter Tino, Ales Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742, oct 2021. doi: 10.1109/tetci.2021.3100641. URL https://doi.org/10.1109%2Ftetci.2021.3100641.

A Appendix

You may include other additional sections here.