Advanced Deep Learning Assignment 4 pwn274

1 The Implementation

For the learning task, I chose to do a binary classification of 0's or 1's in the MNIST data set. This was done by a simple model consisting of a convolutional layer followed by a couple of small linear/fully-connected layers. The code was implemented by use of `PyTorch`. Within two epochs the model gets a validation accuracy of above 99%.

I have implemented the following four interpretability-methods:

```
Integrated Gradients (IG)
Layer-wise Relevance Propagation (LRP)
DeepLIFT
Guided Back Propagation (GBP)
```

All four where calculated using the `captum`-library where lighter pixels are seen as more important.
Examples of their outputs on the two different types of digits can be seen in the following figure:

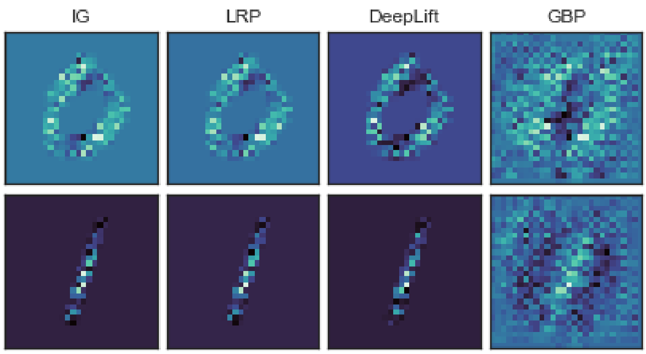

Figure 1: A graphical heat-map of the four different interpretability methods on a 0 and a 1 from the test-set.

2 The Correlations

Running the interpretability algorithms on the full validation set and calculating the Spearman correlation between the four, I find the following results:

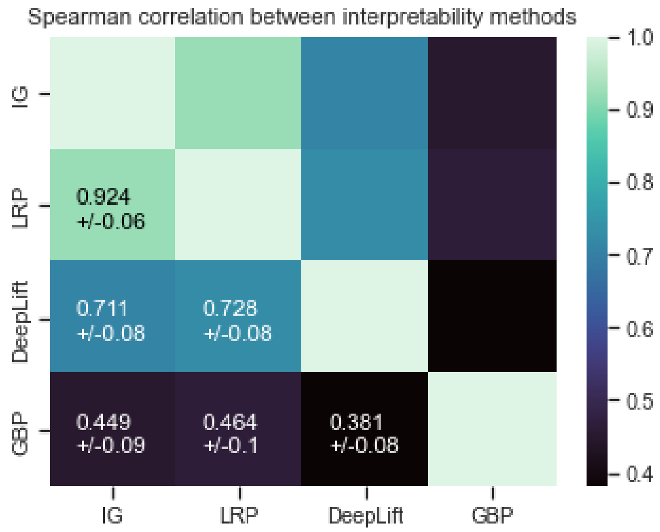

Figure 2: A visualization of the internal empirical Spearman correlation between the four implemented methods. The matrix is of course symmetric and has a diagonal of 1.

The results shown are the empirical means and the uncertainties are the calculated standard deviations.

3 The Explanation

Trying to explain these, I will look into the designs of the algorithms:

IntegratedGradients works by summing up the gradients from a 'baseline' to a given input.[1] The idea of looking at the gradients is also the main idea in Layer-wise Relevance Propagation, I think this is seen in the high correlation between these. DeepLIFT is a SHAP-implementation with the same starting point as Integrated Gradients[2] but introduces the concept of multipliers. This gives it some benefits and makes it different from the former two which can be seen in Figure 2. Guided Back Propagation makes a radical change to the simple gradient-looking algorithms.[3] The base idea is the same, except only the positive weights are looked at. The algorithm only look at what makes "1" a "1" and not what makes it not "0". This might explain the heightened difference in output compared to the other methods.

[1]The original paper
[2]A great, short video explanation of DeepLIFT
[3]Slides For explanation