Learning Important Features Through Propagating Activation Differences

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje

Stanford University

Author Note

Anmol Reddy Mekala,

Computer Science department,

Roll Number: 180050059,

A report for the course project, CS-689 - Machine Learning: Theory & Methods

Abstract

One of the aspects of interpreting the predictions of deep neural networks is attributions of the output prediction to each of the input features. Many of the current feature attribution methods for interpreting the significance of inputs to a neural network suffer from problems with gradient saturation, discontinuities caused with threshold functions etc. Introduced in 2019, the *DeepLIFT* (Deep Learning Important FeaTures) method aims to solve these problems by using a "difference from reference" first proposed in Sundararajan et al., 2017. It combines this along with separating out the positive and negative sums encountered in a pass of a neural network to reveal more dependencies, giving more more expressive attributions. It is able to outperform prior methods both in terms of interpretability and also efficiently (by computing the importance scores in a single backward pass).

In this report, I describe the various methods used for feature attribution, how they were improved until DeepLIFT, and the DeepLIFT method itself. Results of attribution with different methods are also shown. Lastly, the effects of changing the "reference image" hyper-parameter are shown. [1]

Keywords: feature attribution, interpretability, deep neural networks

[1] The code of the experiments is available at https://github.com/molereddy/FeatureAttributions

Learning Important Features Through Propagating Activation Differences

Introduction

Interpretability of neural networks has become one of the major areas of research since the late 2010s. Since the advent of powerful deep leaning models, we've made great strides across various domains like image recognition, speech recognition, natural language processing to name a few. To gain higher and higher accuracies, we have seen state of the art models grow more and more complex in structure and size. Modern deep networks have millions of parameters, a wide variety of layers and perceive complex patterns which need to be interpreted. Interpretability helps debug these models and to decide how to improve them.

Importance scores are among the most intuitive interpretations of neural networks. In computer vision, they are usually called "saliency maps". These are the oldest and most frequently used methods for interpreting the predictions of deep neural networks.

Previous methods

The most primitive methods for example specific explanations of models are perturbation based methods. In these, different areas of the input are shifted at once and the output is noted, e.g. as occlusion maps in Zeiler and Fergus, 2013. These face the drawback of being very computationally inefficient as we have to make a forward pass for each section over the input. Then, gradient-based backpropagation methods were used which developed into the guided backpropagation method (Simonyan et al., 2014).

fireboat picture

simple gradient-based attribution

Figure 1

local perturbations produce a noisy output

Saturation problem in gradient-based methods. The main problem with these methods is the fact that the gradient being calculated is a "local gradient" based on perturbing only at the local neighbourhoods of pixel intensities. Recognition of the object may have happened much before that. This results in attributions as in figure 1. Figure 2 shows us what the problem is: the pixels that decide the image saturate their gradient much ahead of the local gradient that we are calculating. We see in fig 3 that the prediction got most of its impact in the beginning and that behaviour cannot be captured by local gradients.

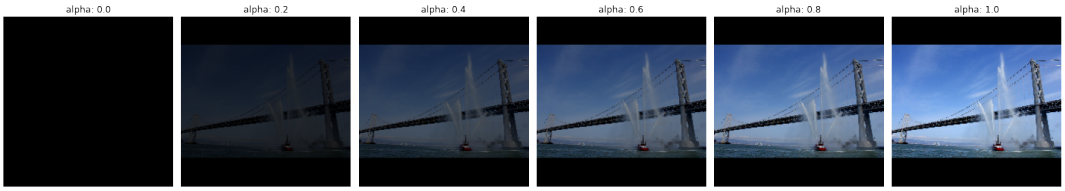


Figure 2

Visualisation of fireboat on varying intensity

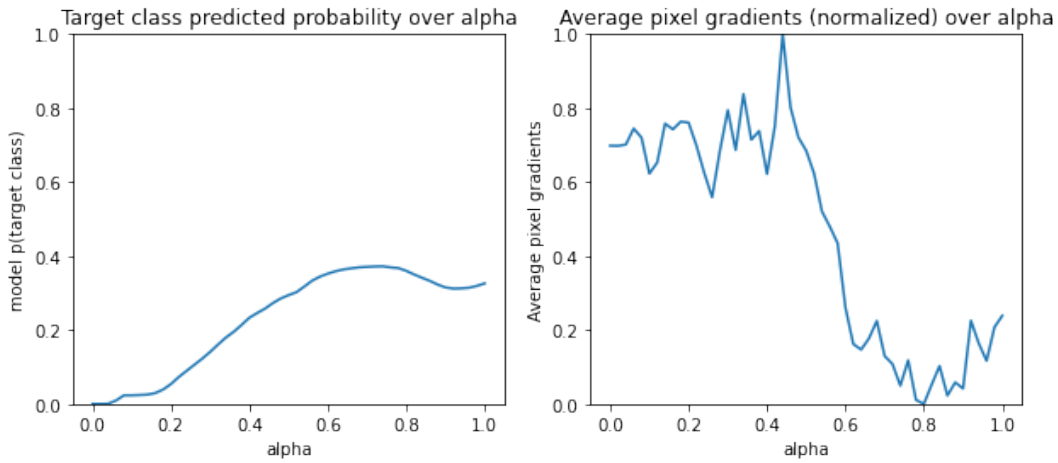


Figure 3

Variation of gradient and prediction at different intensities of the image for the different alphas in fig 2

Thresholding. In these models, functions with a bias as in fig 4 are common. In such situations we have discontinuous gradients. The function always depended on the input, it is only at that particular instance that input is not contributing due to the bias term. We have the recognise this dependence of the function on the input in general and not focus on a particular instance. This problem remains even in methods which improved on simple gradients like GradxInput.

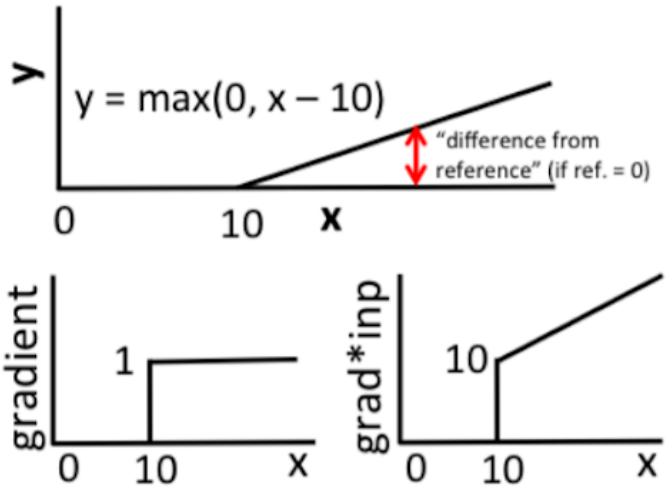


Figure 4

Shows the effect of bias terms giving discontinuous gradients. Taking difference from a reference solves this, as shown

Reference inputs. The solution to both of these problems is to use a reference input and take the difference from it as in fig 4. This difference will always be continuous: solving the thresholding problem. This also solves the saturation problem because we consider the impact of the input "globally" - from reference to actual input rather than a local gradient.

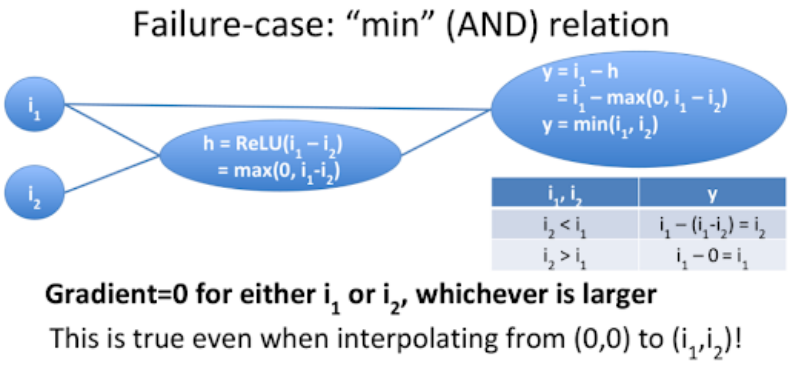
Integrated Gradients method

Proposed in Sundararajan et al., 2017, this method adds the reference input idea and gets the approximated average attribution of each pixel. For example, in fig 3 we use various values along the way (alpha = 0.0, 0.2, ..1 where alpha = 0 is the reference image) and combine and average all the gradients to find the "global" impact. In order to understand the relative importance of pixels, not only in the local neighborhood of pixel intensities, Integrated Gradients redistributes difference of activation of an image and a baseline image. Its attributions sum to the difference in scores between them.

$$\mathsf{IntegratedGrads}_i^{approx}(x) ::=$$
$$(x_i - x_i') \times \Sigma_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m} \times (x - x'))}{\partial x_i} \times \frac{1}{m}$$

This method was put to practical use in Taly et al., 2019 to assist interpretation of retinopathy diagnosis by deep learning models.

Drawbacks: the interpolation requires several gradient calculations which makes it very inefficient. Another problem is that integrated gradients are not expressive enough. This is because in the course of propagation positive and negative signals received by a neuron cancel each other out and this results in us losing information about some dependencies. An example of how info can be lost is shown in fig 6



DeepLIFT method

This method is the next improvement over integrated gradients. Proposed in Shrikumar et al., 2019, this method continues using a reference as in integrated gradients, but adds additional features by back-propagating activation differences Δz instead of interpolated gradients. We express the change in output activation Δy in terms of the input pixel's change from reference Δx. Just as in backpropagation, the weights of neurons and activation function derivatives serve as multipliers. It also separately treats the positive and negative dependencies helping it reveal dependencies missed by other approached. The mathematics of the separation and their description are in the following figures.

$$y = f(x_1, x_2, ...) \quad y^0 = f(x_1^0, x_2^0, ...)$$

$$\sum_{i=1}^{n} C_{\Delta x_i \Delta t} = \Delta t \qquad m_{\Delta x \Delta t} = \frac{C_{\Delta x \Delta t}}{\Delta x}$$

$$m_{\Delta x_i \Delta t} = \sum_j m_{\Delta x_i \Delta y_j} m_{\Delta y_j \Delta t}$$

Figure 5

$m_{\Delta x \Delta t}$ is the multiplier showing the factor by which Δx contributes to Δt. $C_{\Delta x \Delta t}$ is the contribution made. Multipliers are related by a chain rule.

$$\Delta y^+ = \frac{1}{2}\left(f(x^0 + \Delta x^+) - f(x^0)\right) + \frac{1}{2}\left(f(x^0 + \Delta x^- + \Delta x^+) - f(x^0 + \Delta x^-)\right)$$

$$\Delta y^- = \frac{1}{2}\left(f(x^0 + \Delta x^-) - f(x^0)\right) + \frac{1}{2}\left(f(x^0 + \Delta x^+ + \Delta x^-) - f(x^0 + \Delta x^+)\right)$$

$$m_{\Delta x^+ \Delta y^+} = \frac{C_{\Delta x^+ y^+}}{\Delta x^+} = \frac{\Delta y^+}{\Delta x^+}$$

$$m_{\Delta x^- \Delta y^-} = \frac{\Delta y^-}{\Delta x^-}$$

$$\Delta y = \sum_i w_i \Delta x_i.$$

$$\Delta y^+ = \sum_i 1\{w_i \Delta x_i > 0\} w_i \Delta x_i = \sum_i 1\{w_i \Delta x_i > 0\} w_i (\Delta x_i^+ + \Delta x_i^-)$$

$$\Delta y^- = \sum_i 1\{w_i \Delta x_i < 0\} w_i \Delta x_i = \sum_i 1\{w_i \Delta x_i < 0\} w_i (\Delta x_i^+ + \Delta x_i^-)$$

$$C_{\Delta x_i^+ \Delta y^+} = 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^+$$
$$C_{\Delta x_i^- \Delta y^+} = 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^-$$
$$C_{\Delta x_i^+ \Delta y^-} = 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^+$$
$$C_{\Delta x_i^- \Delta y^-} = 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^-$$

Figure 6

We split the contributions into positive and negative portions and have separate multipliers for each

Experiments

In my experiments, I was unable to replicate the DeepLIFT method because of the complications involved in implementing the functions for different kinds of layers. I instead implemented the integrated gradients method and performed experiments on the MNIST and ImageNet datasets (with the GoogLeNet model) with integrated gradients and DeepLIFT pre-implemented from the Captum package. The results here show (7) the comparison of the different methods and also the effect of using different references. MNIST results are in the MNIST.ipynb notebook.

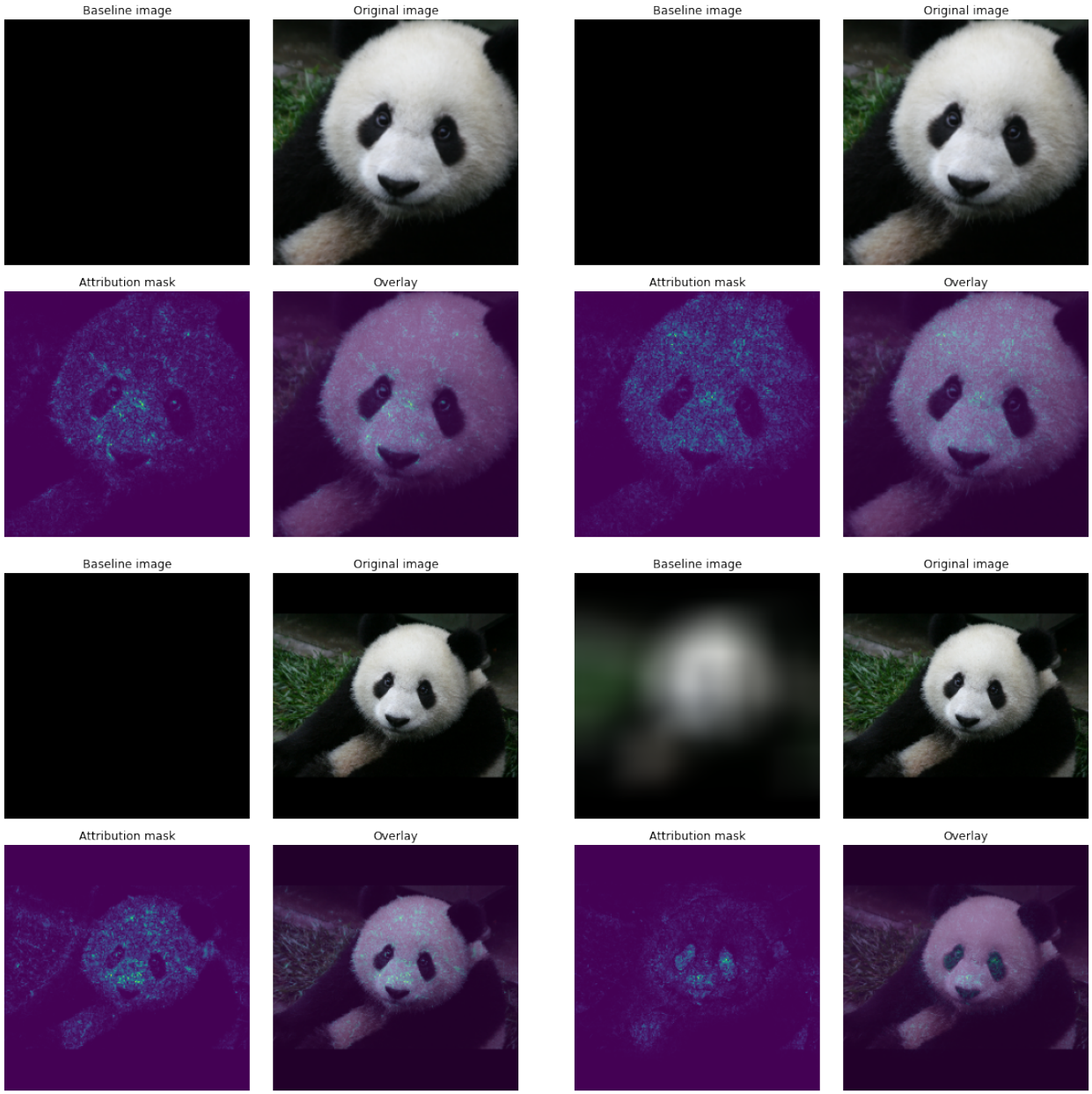


Figure 7

The two images on top show attributions for the 'panda' label in the Integrated Gradients and DeepLIFT methods from left to right respectively. The bottom row shows the effect of using different baseline reference images for attribution

Effect of reference. : The authors propose the reference image to be a kind of neutral baseline which gives no information. For MNIST data, the background is already black so it makes sense to use a black background as reference. But for other datasets where we have a meaningful background (for example, we need the water to recognise a ship in sea), we have to use different backgrounds depending on the situation. According to the DeepLIFT authors, this problem of finding a reference image for our input remains the main open question with this method. Suggestions for methods include using an average from the distribution of all other images in datasets as was used in the DeepShap approach Lundberg and Lee, 2017. I experiment in 7 with a blurred reference. I feel this is intuitive because when we recognise an image, our focus changes from blurred to recognition in the process. So the blurred version has to be neutral. As we see in the figure, usage of this blurred reference helped focus on the eyes, which are the mot distinctive features of a panda.

Conclusion

In this paper, we have seen an efficient method for completely attributing the outputs of a deep learning model in terms of the input values. We noted that DeepLIFT's success lies in its "going deeper": considering dependencies which were cancelled out in other methods and preserving all dependencies.

References

Lundberg, S., & Lee, S.-I. (2017). A unified approach to interpreting model predictions.

Shrikumar, A., Greenside, P., & Kundaje, A. (2019). Learning important features through propagating activation differences.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps.

Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks.

Taly, A., Joseph, A., & Sood, A. (2019). Using a deep learning algorithm and integrated gradient explanation to assist grading for diabetic retinopathy. *Ophthalmology*.

Zeiler, M. D., & Fergus, R. (2013). Visualizing and understanding convolutional networks.