Axiomatic Attribution for Deep Networks

Mukund Sundararajan, Ankur Taly, Qiqi Yan
Google

ICML’17

Presented by Eungyeup Kim

Vision Seminar
15 JUL 2020

Motivation

How do we identify the attribution of input features to the network output?

- A deep-learning based model is considered as a black box.
- Understanding the input-output behavior of the deep network gives us the ability to improve it.
- Attribution techniques based on empirical evaluation cannot differentiate between artifacts from perturbing the data, a misbehaving model and a misbehaving attribution method.

$\Rightarrow$ An attribution method based on axiomatic manner is essential.

Introduction

Integrated Gradients

- This proposed method satisfies all the desirable characteristics, or axioms, for attribution methods.

- This approach only requires an iterative calculation of gradients, resulting in high applicability over several deep networks.

Backgrounds

Baseline
When we assign blame to a certain cause, we implicitly consider the absence of the cause as a baseline for comparing outcomes.

Two fundamental Axioms

1. Sensitivity

: When every input and baseline are different in one feature but have different predictions, the differing feature should be given a non-zero attribution.

2. Implementation Invariance

: Two networks are *functionally equivalent* if their outputs are equal for all inputs, despite having very different implementations.
: Attribution methods should satisfy Implementation Invariance, i.e. the attributions are always identical for two functionally equivalent networks.

⇒ So…existing attribution methods satisfy the axioms mentioned above?
(Gradients, Gradients * inputs, layer-wise relevance propagation(LRP), DeepLift, Deconvolutional Networks, Guided back-propagation…)

Backgrounds

Baseline

When we assign blame to a certain cause, we implicitly consider the absence of the cause as a baseline for comparing outcomes.

Two fundamental Axioms

1. Sensitivity

: When every input and baseline are different in one feature but have different predictions, the differing feature should be given a non-zero attribution.

2. Implementation Invariance

: Two networks are *functionally equivalent* if their outputs are equal for all inputs, despite having very different implementations.

: Attribution methods should satisfy Implementation Invariance, i.e. the attributions are always identical for two functionally equivalent networks.

⇒ So…existing attribution methods satisfy the axioms mentioned above?

(Gradients, Gradients * inputs, layer-wise relevance propagation(LRP), DeepLift, Deconvolutional Networks, Guided back-propagation…)

⇒ Nope!

Backgrounds

Gradients

- Gradients is a reasonable starting point for an attribution method.
- They are invariant to implementation.
- However, they break Sensitivity.

Ex) prediction function may flatten at the input and thus have zero gradient despite the function value at the input being different from that at the baseline.

$\Rightarrow$ Practically, the lack of sensitivity causes gradients to focus on irrelevant features.

Other back-propagation based approaches

- DeepLift, LRP, Deconvnet, Guided back-propagation involve back-propagating the final prediction score through the layers of the network.
- Deconvnet and Guided back-propagation violate Sensitivity.
- DeepLift and LRP tackles the Sensitivity issue by employing a ‘discrete gradient’. In other words, a large, discrete step will avoid flat regions. However, as a result, they suffer from violating Implementation Invariance.

Introduction

Integrated Gradients

This technique combines the Implementation Invariance of Gradients along with the Sensitivity of techniques like LRP or DeepLift.

Suppose we have a function $F : R^n \rightarrow [0,1]$ that represents a deep network. Specifically, let $x \in R^n$ be the input, and $x' \in R^n$ be the baseline input.

We consider the straightline path from the baseline x' to the input x, and compute the gradients at all points along the path. Integrated gradients are obtained by cumulating these gradients.

The integrated gradients along the i^{th} dimension for the input x and baseline x' is defined as follows. Here, $\frac{\partial F(x)}{\partial x_i}$ is the gradient of $F(x)$ along the i^{th} dimension.

$$IntegratedGrads_i(x) ::= (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha$$

If $F : R^n \rightarrow R$ is differentiable almost everywhere, then

$$\sum_{i=1}^{n} IntegratedGrads_i(x) = F(x) - F(x')$$

Introduction

Then how to apply it?

1. Setting Baseline to have near-zero score
- For most deep networks, it is possible to choose a baseline such that the prediction at the baseline is near-zero. ($F(x') \approx 0$)
- In this case, we can distribute the output to the individual input features as individual attributions.

2. Approximation of integrated gradients
- The integral can be efficiently approximated via a summation.

$$IntegratedGrads_i^{approx}(x) ::= (x_i - x_i') \times \sum_{k=1}^{m} \frac{\partial F\left(x' + \frac{k}{m} \times (x - x')\right)}{\partial x_i} \times \frac{1}{m},$$

where m denotes the number of steps in the Riemman approximation of the integral.

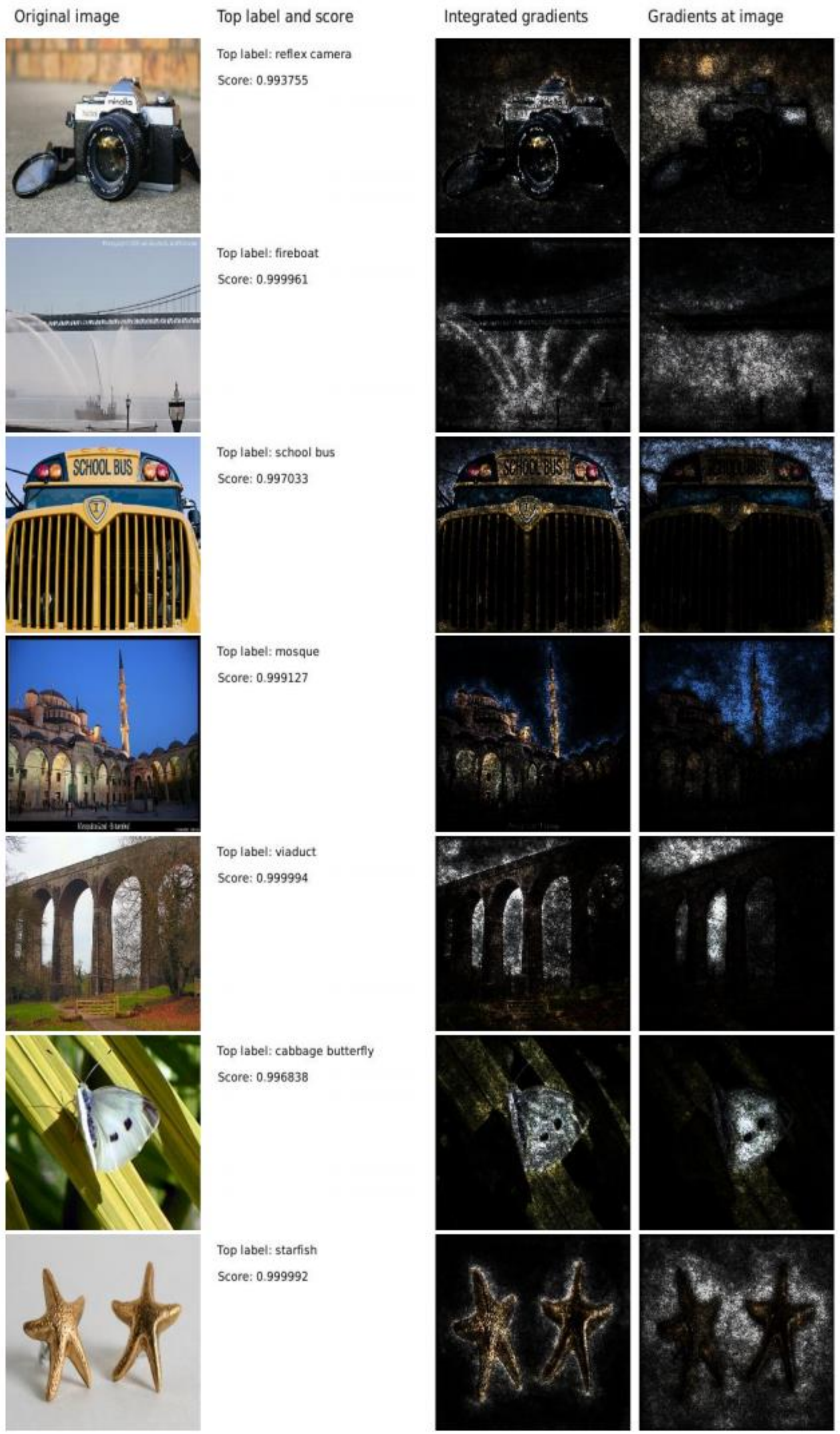
Experiments

Object Recognition Task

- We study feature attribution via GoogleNet trained on ImageNet.

- Integrated gradients can be visualized be aggregating them along the color channel and scaling the pixels in the actual image by them.

- Attribution based on the proposed method is better distributed onto the input pixels, compared to the naïve gradients.



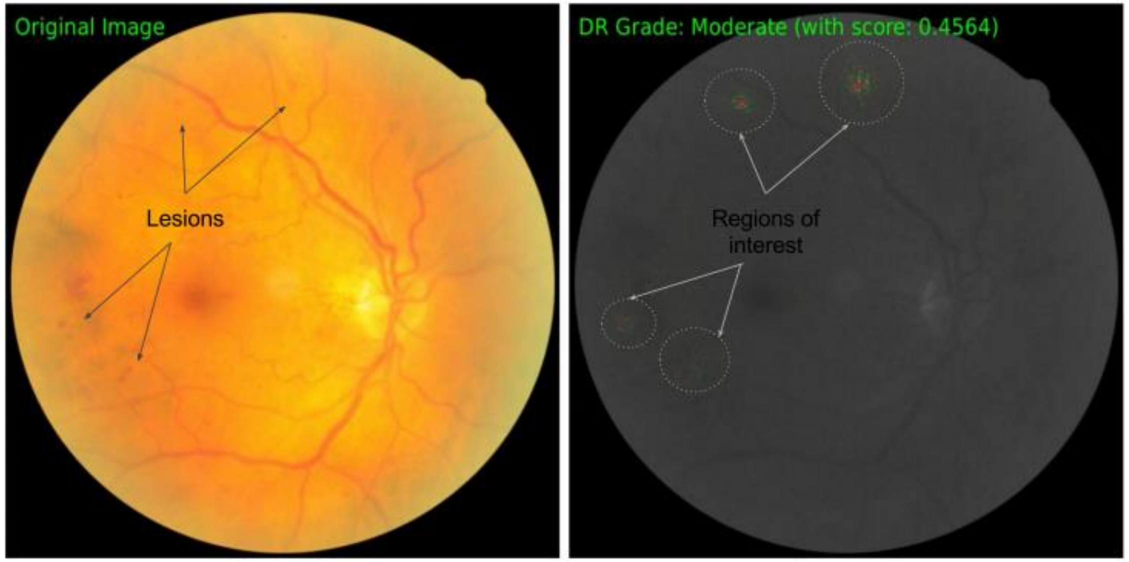
Experiments

Diabetic Retinopathy Prediction

- Diabetic retinopathy (DR) is a complication of the diabetes that affects the eyes.

- Positive attributions are shown in green, and negative are in red channel.

- The interior of the lesions receive a negative attribution while the periphery receives a positive attribution indicating that the network focuses on the boundary of the lesion.



Thank you