Axiomatic Attribution of Neural Networks

Mukund Sundararajan*[1], Ankur Taly*[1], Qiqi Yan*[1]

[1]Google Inc.

Presenter: Arshdeep Sekhon

Motivation

1. Use 'attribution' concept: assigning blame/credit to features
2. Understand the input output behaviour of a network.
3. Interpretability of black box neural networks.

Attribution of Neural Networks

Attribution

Given a function $F : \mathbb{R}^n \rightarrow [0, 1]$, and an input $x \in \mathbb{R}^n$,
Attribution of x relative to baseline $x^{'}$ is
$A_F(x, x^{'}) = (a_1, a_2, \cdots, a_n) \in \mathbb{R}^n$ where a_i is the contribution of x_i to prediction at x i.e. $F(x)$

Challenges for Attribution methods

- Hard to evaluate:

Challenges for Attribution methods

- Hard to evaluate:
- If an attribution method assigns an incorrect attribution: Is the model not doing well (unseen data)?
  Or is the attribution method not good?

Challenges for Attribution methods

- Hard to evaluate:
- If an attribution method assigns an incorrect attribution: Is the model not doing well (unseen data)?
  Or is the attribution method not good?
- Proposed Approach:
  - Introduce two axioms/desirable characteristics that every attribution method should satisfy
  - previous methods do not satisfy atleast one of these two axioms
  - Introduce a new method that satisfies these two axioms

Related Work

- DeepLift
- Layer Wise Relevance Propagation
- Deconvolutional Networks
- Guided Backpropagation
- Gradients: Saliency maps,

Need for a baseline/ choosing a baseline

1. Intuituve: When we assign blame to a certain cause we implicitly consider the absence of the cause for comparison
2. a natural baseline exists in the input space where the prediction is neutral.
3. Example: in object recognition networks, it is the black image. (does not indicate anything)
4. Example: all zero embedding vector for text based tasks indicates nothing.

Axiom 1: Sensitivity

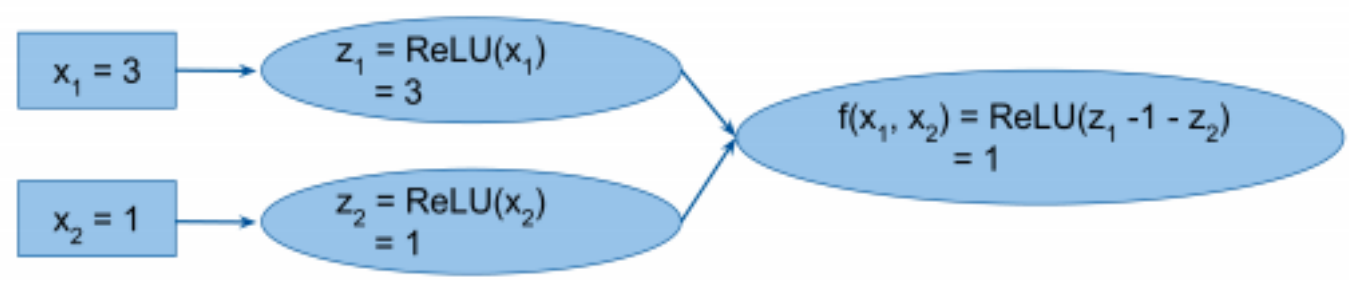
Sensitivity(a)

if for every input and baseline that **differ in one feature** but have **different predictions** then the differing feature should be given a **non-zero attribution**

Gradients violate Axiom 1

- Consider ReLU Network , $f(x) = 1 - ReLU(1 - x)$.
- baseline $x = 0 : F(x) = 0$
- input $x = 2$: $F(x) = 1$
- Gradient $= 0$ at x=2 : function flattens at $x = 2$
- Violate Sensitivity(a): If input is different, the attribution should be non zero.

DeConvNets and Guided Back Propagation violate Axiom 1



Network $f(x_1, x_2)$

- Value of function for fixed $x_1 > 1$ decreases linearly as x_2 increases from 0 to $x_1 - 1$
- Guided Back Propagation/ DeConvNet Rule:

backpropagation: $R_i^l = (f_i^l > 0) \cdot R_i^{l+1}$, where $R_i^{l+1} = \frac{\partial f^{out}}{\partial f_i^{l+1}}$

backward 'deconvnet': $R_i^l = (R_i^{l+1} > 0) \cdot R_i^{l+1}$

guided backpropagation: $R_i^l = (f_i^l > 0) \cdot (R_i^{l+1} > 0) \cdot R_i^{l+1}$

- zero attribution of x_2 because the back-propagated signal received at the node is negative hence not back propagated back

Axiom 2: Implementation Invariance

Functionally Equivalent Networks

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

Implementation Invariance

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

Gradients and Implementation Invariance

Gradients by default are implementation invariant

$$\frac{\partial f}{\partial h}\frac{\partial h}{\partial g} \tag{1}$$

1. If h is some implementation detail of the system, gradient $\frac{\partial f}{\partial g}$ can either be computed directly or through the chain rule.
2. Chain Rule fails for discrete gradients

$$\frac{f(x_1)-f(x_0)}{g(x_1)-g(x_0)} \neq \frac{f(x_1)-f(x_0)}{h(x_1)-h(x_0)}\frac{h(x_1)-h(x_0)}{g(x_1)-g(x_0)} \tag{2}$$

Why implementation invariance?

1. If an attribution method fails to satisfy Implementation Invariance, the attributions are potentially sensitive to unimportant aspects of the models.
2.

For instance, if the network architecture has more degrees of freedom than needed to represent a function then there may be two sets of values for the network parameters that lead to the same function. The training procedure can converge at either set of values depending on the initializtion or for other reasons, but the underlying network function would remain the same. It is undesirable that attributions differ for such reasons.

Proposed Method: Integrated Gradients

1. Combine sensitivity with Implementation Invariance of true gradients
2. a function $F : \mathbb{R}^n \rightarrow [0, 1]$ a deep network
3. input: $x \in \mathbb{R}^n$
4. baseline: $x^{'} \in \mathbb{R}^n$

Integrated Gradients

1. Consider *straightline* path from x to $x^{'}$
2. calculate gradients at each point of these paths
3. Integrated gradients are obtained by cumulating these gradients.
4.

$$IntegratedGrads_i(x) ::= (x_i - x_i^{'}) \times \int_0^1 \frac{\partial F(x^{'} + \alpha \times (x - x^{'}))}{\partial x_i} d\alpha \quad (3)$$

Axiom 3: Completeness

The attributions add up to the difference between the output of F at x and $x^{'}$

Completeness

If $F : \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable almost everywhere:

$$\sum_{i=1}^{n} IntegratedGrads_i(x) = F(x) - F(x^{'}) \tag{4}$$

- If the method satisfies completeness, it clearly satisfies Axiom 1 of sensitivity
- desirable if the networks score is used in a numeric sense

Generalization: Path Gradients

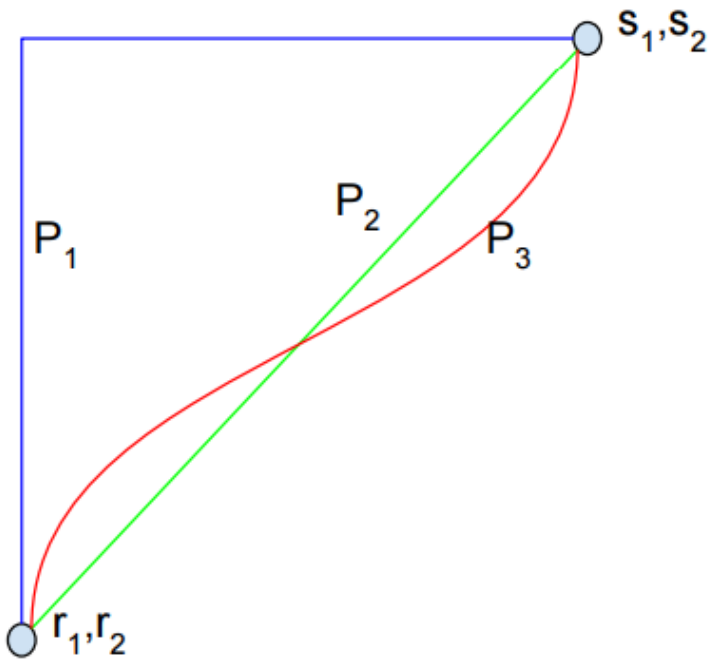


Figure 1. Three paths between an a baseline (r_1, r_2) and an input (s_1, s_2). Each path corresponds to a different attribution method. The path P_2 corresponds to the path used by integrated gradients.

Path Gradients

- let $\gamma = (\gamma_1, \cdots, \gamma_n) : [0, 1] \rightarrow \mathbb{R}^n$ be a smooth function specifying a path in $\mathbb{R}^n$ fron $x^{'}$ to x
- $\gamma(0) = x^{'}$ and $\gamma(1) = x$
- path integrated gradients: integrate along the path $\gamma(\alpha)$ for $\alpha \in [0, 1]$

Path Integrated Gradients

$$PathIntegratedGrads_i^{\gamma}(x) ::= \int_{\alpha=0}^{1} \frac{\partial F(\gamma(\alpha))}{\partial \gamma_i(\alpha)} \frac{\partial \gamma_i(\alpha)}{\partial \alpha} d\alpha$$

Axioms satisfied by Path Gradients

Sensitivity(b):

If a function does not depend on the value of a certain variable, attribution is zero for that variable

Linearity

Linearly compose a function $f_3 = a \times f_1 + b \times f_2$
Attributions should also be the weighted sum of the attributions for f_1, f_2
Intuitively, preerve linearity within network

- Path methods satisfy Implementation Invariance, Sensitivity(b), Linearity, and Completeness

[add citation]

Integrated Gradients are symmetry preserving

Symmetry preserving inputs

Two input variables are symmetry preserving if swapping them does not change the output

$$F(x, y) = F(y, x) \forall x, y \tag{5}$$

Symmetry preserving attribution methods

For all inputs that have identical values for symmetric variables and baselines that have identical values for symmetric variables, the symmetric variables receive identical attributions.

Proof:

1. non straightline path $\gamma : [0, 1] \rightarrow \mathbb{R}^n$
2. Without loss of generality, there exists $t_0 \in [0, 1]$ such that for two dimensions i, j, $\gamma_i(t_0) > \gamma_j(t_0)$.
3. (t_1, t_2) is the maximum real open interval containing t_0 s.t. $\gamma_i(t) > \gamma_j(t) \forall t \in (t_1, t_2)$

Proof: Integrated Gradients are Symmetry Preserving

1. Define $f : x \in [0,1]^n \rightarrow \mathbb{R}$
2. 0 if $min(x_i, x_j) \leq a$
3. $(b-a)^2$ if $max(x_i, x_j) \geq b$
4. otherwise $(x_i - a)(x_j - a)$
5. compute the attributions of f at $x = (1, ..., 1)_n$in with baseline $x_0 = (0, ..., 0)_n$.
6. For $t \notin [t1, t2]$ the function is a constant, zero attribution to all
7. For $t \in [t1, t2]$ the integrand of attribution of f is $\gamma_j(t) - a$ to x_i and $\gamma_i(t) - a$ to x_j
8. latter is always strictly larger by choice of the interval.
9. contradiction: x_j gets a larger attribution than x_i: *contradiction*

Computing Integrated Gradients

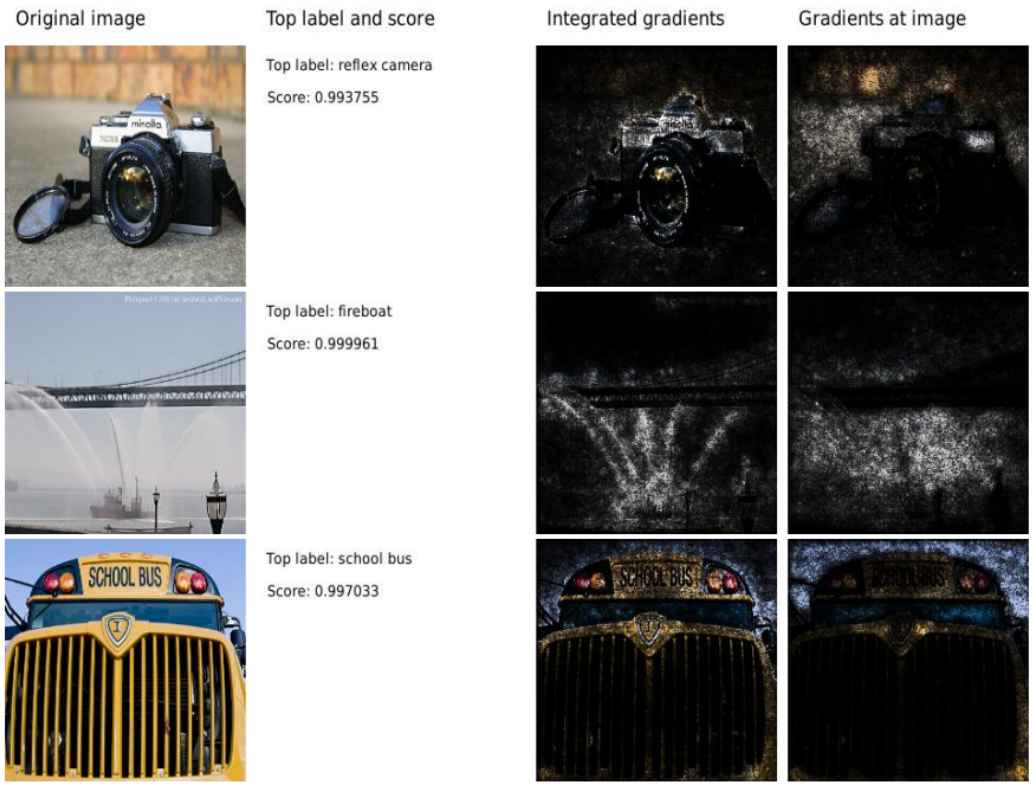
Riemman Sum approximation of an integral:

$$S = \sum_{i=1}^{m} f(x_i^*)\Delta(x_i) \tag{6}$$

$$IntegratedGrads_i^{approx}(x) = (x_i - x_i^{'}) \sum_{k=1}^{m} \left(\frac{\partial F(x^{'} + \frac{k}{m} \times (x - x^{'}))}{\partial x_i}\right) \times \frac{1}{m} \tag{7}$$

Experiment 1: Object Recognition Network

- using the GoogleNet architecture and trained over the ImageNet object recognition
- The gradients are computed for the output of the highest-scoring class with respect to pixel of the input image dataset.



Experiment 2: Question Classification

- What type of question: For example, yes/no or a date?
- triggers: 'what'/'when'

how many townships have a population above 50 ? [prediction: NUMERIC]
what is the difference in population between fora and masilo [prediction: NUMERIC]
how many athletes are not ranked ? [prediction: NUMERIC]
what is the total number of points scored ? [prediction: NUMERIC]
which film was before the audacity of democracy ? [prediction: STRING]
which year did she work on the most films ? [prediction: DATETIME]
what year was the last school established ? [prediction: DATETIME]
when did ed sheeran get his first number one of the year ? [prediction: DATETIME]
did charles oakley play more minutes than robert parish ? [prediction: YESNO]

Experiment 3: Neural Machine Translation

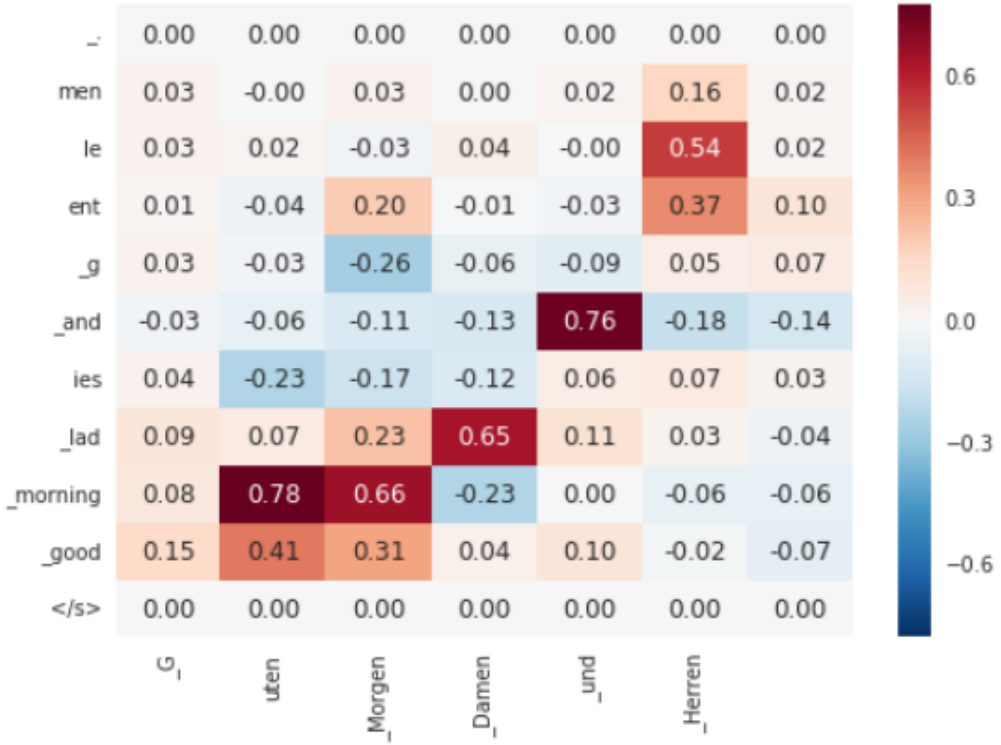


Figure 5. **Attributions from a language translation model.** Input in English: "good morning ladies and gentlemen". Output in German: "Guten Morgen Damen und Herren". Both input and output are tokenized into word pieces, where a word piece prefixed by underscore indicates that it should be the prefix of a word.

Conclusions

1. integrated gradients approach that attributes the prediction of a deep network to its inputs
2. Easy to implement
3. clarifies desirable features of an attribution method using an axiomatic framework
4. does not address the interactions between the input features or logic of network