A General Taylor Framework for Unifying and Revisiting Attribution Methods

Huiqi Deng ¹, Na Zou ², Mengnan Du ², Weifu Chen ¹, Guocan Feng ¹, Xia Hu ²

Abstract—Attribution methods provide an insight into the decision-making process of machine learning models, especially deep neural networks, by assigning contribution scores to each individual feature. However, the attribution problem has not been well-defined, which lacks a unified guideline to the contribution assignment process. Furthermore, existing attribution methods often built upon various empirical intuitions and heuristics. There still lacks a general theoretical framework that not only can offer a good description of the attribution problem, but also can be applied to unifying and revisiting existing attribution methods. To bridge the gap, in this paper, we propose a Taylor attribution framework, which models the attribution problem as how to decide individual payoffs in a coalition. Then, we reformulate fourteen mainstream attribution methods into the Taylor framework and analyze these attribution methods in terms of rationale, fidelity, and limitation in the framework. Moreover, we establish three principles for a good attribution in the Taylor attribution framework, i.e., low approximation error, correct Taylor contribution assignment, and unbiased baseline selection. Finally, we empirically validate the Taylor reformulations, and reveal a positive correlation between the attribution performance and the number of principles followed by the attribution method via benchmarking on real-world datasets.

1 INTRODUCTION

Deep Neural Networks (DNNs) have achieved great success in many real-world applications. The tremendous parameter space enhances the function approximation ability and hence improves the model performance. However, they also compromises over the model transparency, making it difficult to interpret the model decision. The concerns about the interpretability of DNNs have hampered their further applications, especially in high-stake applications such as automatic driving and AI healthcare. Hence developing model interpretation to promote a trustworthy AI is extremely important and has drawn an increasing attention recently [1].

Attribution methods have become an effective computational tool in understanding the behavior of machine learning models, especially Deep Neural Networks (DNNs) [1], [2], [3]. They uncover how machine learning models make a decision by calculating the contribution score of each input feature to the final decision. For example, in image classification, the attribution methods infer the contribution of each pixel to the predicted label for a pre-trained model, and usually create heatmaps to visualize the contributions.

Although several attribution methods [4] have been proposed recently, the attribution problem is actually not well-defined. Sundararajan [5] roughly defines “the attribution of input $x$ relative to a baseline point $\tilde{x}$” as a vector $a = [a_1, \ldots, a_n]$, where $a_i$ denotes the contribution of feature $x_i$ to the prediction. Such description is uninformative to understanding attribution problem, which lacks a concrete guide to the logic of contribution assignment process. Moreover, existing attribution methods are based on different heuristics and have very limited theoretical understanding and support. For instance, Layer-wise Relevance Propagation (LRP) evaluates the contribution of each neuron at the lower layer to a non-linear neuron at the upper layer based on the proportion of lower neuron’s linear combination value [6]. In addition, the saliency map of smooth gradients [7] obtains significantly improved performance than gradient just by averaging the gradients of neighbors. The rationales behind these methods are perplexing.

Hence, it’s highly desirable to not only deepen the understanding of attribution problem, but also conduct a comprehensive exploration and investigation to those various heuristic attribution methods. Specifically, the following important questions for attribution methods need theoretical investigation: Rationale—What model behaviors do these attribution methods actually reveal; Fidelity—How much can decision making process be attributed in these attribution methods; Limitations—Where these attribution methods may fail.

While some attempts have been made to partially answer the questions by unifying several attribution methods as additive feature attribution [5], multiplying a modified gradient with input [8], or first-order Taylor expansion [4], the problems are still not addressed well due to two challenges. The first challenge (Ch1) is to our knowledge, none of them could offer a good description to the attribution problem. The second challenge (Ch2) stems from the fact that it is very difficult to propose a general framework to unify most existing attribution methods, because these methods are based on various heuristics.

In this paper, we address the aforementioned problems by proposing a general Taylor attribution framework, which not only offers a good description to the attribution problem (section 5), but also unifies fourteen mainstream attribution methods into the framework by Taylor reformulations (section 4). The basic idea behind the proposed framework is to attribute a Taylor approximation function of DNNs,
We propose a general Taylor attribution framework, which offers a good description to the attribution problem. The framework provides an insight into the logic of contribution assignment process.

Fourteen mainstream attribution methods are unified into the proposed framework by theoretical reformulations.

- Based on unified Taylor reformulations, we revisit existing attribution methods in terms of their rationale, fidelity, and limitations. We also accordingly establish three principles for a good attribution.
- We empirically validate the Taylor reformulations, and reveal the relationship between attribution performance and the three principles on MNIST and Imagenet.

## 2 RELATED WORK

In this section, we firstly provide an overview of existing attribution methods. Then, we introduce the related works which pay attention to understanding and unifying these existing attribution methods in details.

### 2.1 Existing attribution methods

Attribution is an effective computational tool in locally interpreting the behavior of machine learning models, especially DNNs. Recently, there are a number of attribution methods developed to infer the contribution score of each input feature to the final prediction for a given input sample. Saliency maps are usually created to visualize the contribution score. We roughly categorize these attribution methods into local attribution explanation approach and global attribution explanation approach.

Local attribution explanation approach focuses on the sensitivity of the difference of the output neuron w.r.t each input neuron, i.e., how the output of the network changes for infinitesimally small perturbations around the original input. Gradient [9] calculates the sensitivity, which masks out the negative neurons of bottom data via the forward ReLu at the ReLu layer. To improve the saliency map quality, smooth gradients [7] produces an attribution vector by averaging the gradients of neighbor samples, which is generated through adding Gaussian noise to the original given sample. Deconvnet [10] aims to map the output neuron back to the input pixel space. To keep the neuron’s size and non-negative property, they resort to the transposed filters and backward ReLu, which masks out the negative neurons of the top gradients. The Guided Back-propagation (GBP) [11] combines Gradients and Deconvnet, considering both forward relu and backward relu. As a result, GBP could significantly improve the visual quality of visualizations.

Global attribution explanation approach directly analyzes or decomposes the output difference between the input and the selected baseline. Gradient*Input [12] calculates the attributions by multiplying the gradient with the input pixel space. To keep the neuron’s size and non-negative property, they resort to the transposed filters and backward ReLu, which masks out the negative neurons of the top gradients. The Guided Back-propagation (GBP) [11] combines Gradients and Deconvnet, considering both forward relu and backward relu. As a result, GBP could significantly improve the visual quality of visualizations.

Global attribution explanation approach directly analyzes or decomposes the output difference between the input and the selected baseline. Gradient*Input [12] calculates the attributions by multiplying the gradient with the original input, to improve the sharpness of saliency maps. Grad-CAM focuses on interpreting the classification module of convolutional neuron networks (CNNs). It captures the importance of each feature channel at the top convolutional layer, which conducts global average pooling to the gradients of the output neuron w.r.t each feature map. Then Grad-CAM obtain a coarse attribution by multiplying the importance with these feature maps. Occlusion-1 [10] and Occlusion-patch [13] observes the changes of the output induced by occluding each input pixel or patch. Layer-wise Relevance Propagation (LRP) decomposes the value of output neuron in a layer-wise manner. Specifically, it
recursively decomposes the relevance score of a neuron at the upper layer to the neurons at the lower layer, according to the corresponding proportion in the linear combination. DeepLIFT Rescale rule [14] adopts a similar linear rule to \(\epsilon\)-LRP, while it assigns the difference between the output and a baseline output in terms of the difference between the input and a pre-set baseline input, instead of merely considering the output value. Integrated Gradients [5], corresponds to Aumann-Shapley, decomposes the output difference by integrating the gradients along a straight path interpolating from input sample to the baseline. In addition, Shapley value [3] has become a popular attribution method, which calculates an average marginal contribution of each feature across all possible feature subsets. Shapley value is characterized by a collection of desirable properties, e.g., local accuracy, missingness, and consistency.

Moreover, some variants of above global explanation methods have been proposed recently. Generally, they adopt two strategies to improve the attribution results: i) disentangling the contributions from positive and negative terms. For example, DeepLIFT RevealCancel [14] separately considers the overall marginal impact of the positive terms and negative terms. Layer-wise relevance propagation \(\alpha \beta\) rule (LRP-\(\alpha \beta\)) [6] decompose \(\alpha \) the overall effects to the positive terms and \(\beta\) times to the negative terms, where \(\alpha \) and \(\beta\) satisfy \(\alpha - \beta = 1\) to ensure the completeness. Deep Taylor [15] has been demonstrated as a special case of LRP-\(\alpha \beta\) when \(\alpha = 1\) and \(\beta = 0\). ii) averaging over multiple baselines to reduces the probability that the attribution is dominated by a specific baseline. Such strategy can be integrated into most attribution methods. The corresponding version of Integrated Gradients (Expected Gradients), DeepLIFT (Expected DeepLIFT), and Shapley value (Deep Shapley) have been shown significantly improve the interpretation performance.

In this paper, we mainly focus on the global attribution explanation approach, because they usually analyze or decompose the output difference between input and baseline and can be represented as a function of such difference. Therefore it’s natural to reformulate these methods into the proposed Taylor attribution framework.

### 2.2 Understanding and unifying attribution methods

There are a few works on understanding the theoretical grounds of some attribution methods that are often designed heuristically. \(\epsilon\)-LRP [16] and LRP-\(\alpha \beta\) [4] are reformulated as a first-order Taylor decomposition. Moreover, Deconvnet and Guided BP have been theoretically proved [17] that they are essentially constructing (partial) recovery to the input, which is unrelated to decision making.

Some efforts have been devoted to unifying existing attribution methods. LIME, LRP, DeepLIFT, and Shapley value are unified under the framework of additive feature attribution [3]. Some gradient-based attribution methods including Gradient*Input, \(\epsilon\)-LRP, DeepLIFT and Integrated Gradient, are unified as multiplying a modified gradient with \(a\) [8]. Several equivalence conditions are given. In addition, several methods are summarized as first-order Taylor decomposition on different baseline points [4].

To our knowledge, this is the first work to leverage high-order Taylor decomposition and interactive effects to formally define the attribution problem and unify the majority of existing attribution methods.

### 3 Taylor Attribution Framework

In this section, we propose a Taylor attribution framework to deepen the intrinsic understanding of attribution problem. Given a pre-trained DNN model \(f\) and an input sample \(x = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n\), the attribution problem aims to infer the contribution of each feature \(x_i\) to the prediction \(f(x)\). Existing attribution methods usually select a baseline point \(\bar{x} = [\bar{x}_1, \ldots, \bar{x}_n]^T\) to represent a reference state, then the output difference between the input and baseline point \(\Delta f = f(\bar{x}) - f(x)\), can be considered as the influence caused by the input difference \(\Delta = \bar{x} - x\). Hence, attribution can be seen as the process assigning the output difference \(\Delta f\) to each feature \(x_i\) according to their corresponding input difference \(\Delta_i = \bar{x}_i - x_i\). However, there are infinite possible cases to decompose a scalar to a \(n\)-dimensional vector. No work has provided a guidance to the concrete logic in the contribution assignment process, i.e., which assignment is logical and reasonable.

To offer a good description and depiction to the attribution problem, we resort to Taylor decomposition theory and
propose a general Taylor attribution framework. Specifically, the basic idea is that we conduct the attribution for the Taylor approximation function $g$ of DNN model, instead of directly attributing DNN itself. The idea is doable due to two aspects. Firstly, Taylor expansion can sufficiently approximate DNN model $f$ so that the above two attributions are approximately equivalent. Secondly, the Taylor expansion function is polynomial, which is easier to analyze how to assign the contribution intuitively.

Assume $f$ is differentiable, the Taylor expansion of $f$ expanded at input sample $x$ is

$$f(\tilde{x}) - f(x) = g_K(x, \Delta) + \epsilon_K(\tilde{x}),$$

where $g_K(x)$ is the $K$-order Taylor expansion function of $f$. $g_K(x, \Delta)$ is the function value of $g_K(x)$ at baseline $\tilde{x}$, and $\epsilon_K(x, \Delta)$ is the approximation error between $f(x)$ and $g_K(x, \Delta)$ at point $\tilde{x}$. The left side of equation, $f(\tilde{x}) - f(x)$, represents the output difference, which can be considered as the influence of input difference $\Delta$. We need to answer: how to decompose the effect according to the input difference $\Delta_i$ of each feature. It’s difficult to decompose directly the effect $\Delta f$ due to the complexity of $f$. As $g_K(x, \Delta)$ is an approximation of the output difference, we instead decompose $g_K(x, \Delta)$ into a attribution vector $a = [a_1, \ldots, a_n]^T$, where $a_i$ denotes the attribution score of feature $x_i$ to $g(x, \Delta)$.

An overview of Taylor attribution framework is illustrated in Figure 2. For convenience, we summary the descriptions of main symbols in this paper in Table 1.

### 3.1 First-order Taylor attribution

The first-order Taylor expansion function $g_1(x, \Delta)$ is

$$g_1(x, \Delta) = f_x^T \Delta = \sum_i f_{x_i} \Delta_i,$$

where $f_{x_i}$ denotes the derivative of $f$ with respect to $x_i$. The linear approximation function in first-order Taylor expansion, $g_1(x, \Delta)$, is additive across features and can be easily decomposed. It’s obvious that $f_{x_i} \Delta_i$ quantifies the contribution of feature $x_i$, i.e.,

$$a_i = f_{x_i} \Delta_i.$$

1. Noted that although the deep relu network is not differentiable such that Taylor expansion is not applicable, networks with softplus activation (approximation of relu) can be used to provide an insight to the rationale behind relu net.

### 3.2 Second-order Taylor attribution

The second-order Taylor expansion has a smaller approximation error $\epsilon$ than the first-order one, so that it is expected more faithful to the model $f$. The second-order Taylor expansion function $g_2(x, \Delta)$ is given by

$$g_2(x, \Delta) = \frac{1}{2} f_x^T \Delta + \frac{1}{3} f_{w^T} \Delta^2,$$

where $H_x$ is the Hessian matrix, i.e., second-order partial derivative matrix, of $f$ at $x$. We denote the first-order and second-order Taylor terms as $T^\alpha$ and $T^\beta$, respectively.

The second-order Taylor expansion function $g_2(x, \Delta)$ is indistinct in determining feature contributions compared with first-order one due to the Hessian matrix. To make the attribution more clear, we decompose $H_x$ into two matrices, an independent matrix $H_x^a$ and an interactive matrix $H_x^b := H_x - H_x^a$. Here $H_x^a$ is a diagonal matrix composed of the diagonal elements in $H_x$, which describes the second-order isolated effect of features, and $H_x^b$ represents the interactive effect between features. $g_2(x, \Delta)$ could be rewritten as the sum of first order terms $T^\alpha$, second-order independent terms $T^\beta$, and second-order interactive terms $T^\delta$,

$$g_2(x, \Delta) = f_x^T \Delta + \frac{1}{2} f_x^T H_x^a \Delta + \frac{1}{2} f_x^T H_x^b \Delta.$$

Accordingly, the attribution of $g_2(x, \Delta)$ to $x_i$ should be

$$a_i = a_i^\alpha + a_i^\beta + a_i^\delta,$$

where $a_i^\alpha$, $a_i^\beta$, and $a_i^\delta$ represent the assigned contributions from $T^\alpha$, $T^\beta$, and $T^\delta$, respectively. The contributions from independent terms $T^\alpha$ and $T^\beta$ can be clearly identified as

$$a_i^\alpha = T_i^\alpha, \quad a_i^\beta = T_i^\beta = \frac{1}{2} f_{x_i} \Delta_i^2,$$

where $T_i^\alpha$ and $T_i^\beta$ denote the first-order terms and second-order independent terms of feature $x_i$, respectively.

The difficulty lies on how to assign the contribution from interactive terms $T^\delta$. We propose to handle it by following an intuition behind: the assignment $a_i^\delta$ is from $T^\delta$, and should be the sum of assignments from each interactive effect involving feature $x_i$,

$$a_i^\delta = \sum_{j \neq i} a_i^{\delta_{x_i, x_j}} = \sum_{j \neq i} w_{i,j} T_{i,j}^{\delta_{x_i, x_j}},$$

where $T_{i,j}^{\delta_{x_i, x_j}} = f_{x_i, x_j} \Delta_i \Delta_j$ denotes the second-order interactive terms corresponding to feature $x_i$ and $x_j$, weight
\( w^{(i,j)} \) characterizes the assignment of the interactive terms to \( x_i \), and \( a^{\beta}_{\{x_i,x_j\}} \) is the attribution from \( T^{\beta}_{\{x_i,x_j\}} \).

The determination of the assignment weight \( w^{(i,j)} \) is complicated and depends on specific cases. However, it's considered that the interactive terms of two features should be only attributed to these two features. Consider the interactive terms between \( x_i \) and \( x_j \), the assignment should satisfy \( a^{\beta}_{\{x_i,x_j\}} + a^{\beta}_{\{x_j,x_i\}} = T^{\beta}_{\{x_i,x_j\}} \), i.e., \( w^{(i,j)} = w^{(j,i)} = 1 \). For example, as shown in the Taylor reformulation in section 4, Integrated Gradients assigns the interactive terms according to the order of features. Because the order of \( x_i \) and \( x_j \) in second-order interactive terms (i.e., \( f_{x_ixj} \Delta_i \Delta_j \)) are both 1, so the term are equally assigned to \( x_i \) and \( x_j \). That is, \( w^{(i,j)} = w^{(j,i)} = \frac{1}{2} \) in Integrated Gradients.

### 3.3 High-order Taylor attribution

The analysis on second-order expansion can be naturally extended to high-order expansion where \( K > 2 \). Let \( T^0 \) denote all high-order expansion terms, including second-order expansion terms. The high-order Taylor expansion function at \( x \) is

\[
g_R(x, \Delta) = T^0 + T^{\gamma^d} + T^{\gamma^i},
\]

where \( T^{\gamma^d} \) and \( T^{\gamma^i} \) denote high-order independent and interactive terms, respectively.

Analogously to the second-order case, the attribution of feature \( x_i \) in high-order expansion is given by

\[
a_i = a_i^{\gamma^d} + a_i^{\gamma^i},
\]

where \( a_i^{\gamma^d}, a_i^{\gamma^i} \) represent the assigned contributions from \( T^{\gamma^d} \) and \( T^{\gamma^i} \), respectively. The attribution from first-order term and high-order independent term is clear,

\[
a_i^0 = T_i^0, \quad a_i^{\gamma^d} = T_i^{\gamma^d},
\]

where \( T_i^{\gamma^d} \) represent the high-order independent terms of feature \( x_i \). The attribution from interactive terms, \( a_i^{\gamma^i} \), consists of all assignments from interactive terms involving \( x_i \),

\[
a_i^{\gamma^i} = \sum_{\substack{\Delta \in \Delta \setminus \{x_j\} \atop x_j \in A}} a_{\Delta}^{\gamma^i}, x_i \in A,
\]

where \( a_{\Delta}^{\gamma^i} \) denotes the attribution from interactive terms corresponding to features in the feature subset \( A \). Note that the interactive terms \( T_i^{\gamma^i} \) should be only assigned to the features in the subset \( A \), i.e.,

\[
\sum_{\Delta \subseteq A} a_{\Delta}^{\gamma^i} = T_i^{\gamma^i}.
\]

Based on the analysis, we give a definition for how to correctly assign Taylor contribution.

**Definition 1.** A Taylor attribution has a correct Taylor contribution assignment if the attribution is given by,

\[
a_i = T_i^0 + T_i^{\gamma^d} + a_i^{\gamma^i},
\]

and the assignment from interactive terms satisfies, \( \forall A \)

\[
\sum_{\Delta \subseteq A} a_{\Delta}^{\gamma^i} = T_i^{\gamma^i}.
\]

In brief, Eq. [1] indicates that the Taylor first-order and high-order independent terms of feature \( x_i \) should be assigned to \( a_i \), and part of Taylor interactive terms \( T_i^{\gamma^i} \) involving feature \( x_i \) should be allocated to \( a_i \). Eq. [2] requires that the interactive terms of features in subset \( A \) should be and only be attributed to the features in subset \( A \). It’s worthy noting that the high-order term \( T_i^{\gamma^i} + a_i^{\gamma^i} \) can be omitted, if the first-order Taylor expansion can approximate the model \( f \) sufficiently.

### 3.4 The selection of baseline point

From the Taylor attribution framework, the attribution of feature \( x_i \) could be seen as a polynomial function of \( \Delta_i \) (i.e., \( \Delta_i = x_i - \tilde{x}_i \)), and hence it highly depends on \( \tilde{x}_i \). Given a constant vector baseline \( \tilde{x} = c \) as many attribution methods did, the attribution of feature whose value is far from \( c \) may be overestimated due to a large \( \Delta_i \), while the attribution of feature whose value is close to \( c \) may be underestimated even if it is important to the decision making process. Such different attributions are a bias in many tasks. For example, in image classification, it’s unreasonable to attribute according to the value of features (i.e., pixel values). Specifically, given a black image as baseline, pixels in white color have a large \( \Delta_i \) close to 255, while pixels in black color have a small \( \Delta_i \) close to 0. Correspondingly, the attribution methods will biasedly highlight white pixels while neglecting black pixels even if black pixels make up the object of interest. Hence the selection of baseline point \( \tilde{x} \) plays a significant role.

Baseline point is used to represent an “absence” of a feature, by which the attribution methods calculate how much the output of the model would decrease considering the absence of the feature [15]. Hence, it’s expected that the output of baseline point has a significant decrease. Moreover, to avoid incorporating aforementioned bias into the attribution process, attribution methods should choose an unbiased baseline which satisfies there is no big differences among \( \Delta_i \) of different features. That is, \( \Delta_i \) should be similar to \( \Delta_j \) for random two feature dimensions. One option is setting \( \Delta \) as a constant vector \( c \) and its corresponding baseline is \( \tilde{x} = x + c \). Such baselines indeed solve the bias issue, however they usually don’t make a difference to the output of the model. Another alternative is to sample the input difference \( \Delta \) from distributions (e.g., uniform and Gaussian distributions) with zero mean and small variance, to ensure a small difference among \( \Delta_i \) of different features. In addition, the biases can be further neutralized by averaging multiple baselines whose \( \Delta \) are sampled from such distributions. This strategy reduces the probability that the attribution is dominated by a specific baseline, which is prone to be biased. This may explain why SmoothGrad [7] and Expected Gradients [19] will success with small Gaussian variance level while fail with large variance.

### 4 Unified Taylor Reformulations

The proposed Taylor attribution framework is very general, and it can unify attribution methods based on the analysis of output difference. These attribution methods assign/decompose the output difference \( \Delta f \) between input and baseline point to each input feature. In these attribution methods, the attribution is performed by a function of output difference. Moreover, such output difference can be approximately represented as the sum of Taylor terms by Taylor decomposition. Therefore, the attribution can be
unified into our framework, i.e., the attribution could be reformulated as a function of the Taylor terms.

In this section, we will unify fourteen mainstream attribution methods into the proposed Taylor framework by Taylor reformulations, and all the proofs of theorems are in the Appendix. This section is organized as follows. Firstly, we discuss about eight basic versions of attribution methods, which are Gradient*Input [12], Grad-CAM [20], Occlusion-1 [10], Occlusion-patch [13], Integrated Gradients [9], DeepLIFT Rescale [14], $\epsilon$-LRP [6] and Shapley value [3]. According to whether the method considers feature interactions, we categorize them into two types. Secondly, we study the variants which disentangle the contributions from positive and negative terms. Specifically, this part includes DeepLIFT Rescale (DeepLIFT ReveaLCancel [14]) and $\epsilon$-LRP (LRP-$\alpha\beta$ [6] and Deep Taylor [15]). Thirdly, we reformulate the variants averaging over multiple baselines, which are Expected Gradients [19], Expected DeepLIFT, and Deep Shapley [3].

4.1 Without feature interaction

In this subsection, we demonstrate that after Taylor reformulations, the following five attribution methods don’t consider feature interactive terms (completely).

4.1.1 Gradient*Input

The attribution in Gradient*Input [12] is calculated by multiplying the partial derivatives (of output w.r.t input) with the input, i.e., $a_i = f_{x_i}(x)x_i$. It’s easy to obtain Theorem 1.

**Theorem 1.** Gradient*Input can be reformulated as a first-order Taylor attribution w.r.t the baseline point $\tilde{x} = 0$,

$$a_i^{GI} = T_i^\alpha.$$ 

4.1.2 $\epsilon$-LRP

$\epsilon$-LRP [6] proceeds in a layer-wise back-propagation fashion. Use $x_i^{(l)}$ and $x_j^{(l+1)}$ to denote the neuron $i$ at l-th layer and the neuron $j$ at $(l + 1)$-th layer, respectively, and $x_j^{(l+1)} = \sigma(\sum_i w_{ji}x_i^{(l)} + b_j)$. Here $w_{ji}$ is the weight parameter, $b_j$ is the additive bias, and $\sigma$ is a non-linear activation function. $\epsilon$-LRP recursively decomposes the relevance score of $j$-th neuron at $(l+1)$-layer to the neurons at $l$-th layer, according to the proportion of weighted impacts. Then the attribution of $i$-th neuron from $j$-th neuron at $(l+1)$-layer is,

$$a_{ij}^{(l)} = \frac{z_{ij}^{(l)}}{(\sum_i z_{ij}^{(l)} + b_j) + \epsilon \cdot \text{sign}(\sum_i z_{ij}^{(l)} + b_j)} a_{ij}^{(l+1)}. \tag{3}$$

where $z_{ij}^{(l)} = w_{ji}x_i^{(l)}$ is the weighted impact of $x_i^{(l)}$ to $x_j^{(l+1)}$. $a_{ij}^{(l+1)}$ is the total relevance score of neuron $j$ and will be assigned to features at l-th layer. Here $\epsilon$ is a small quantity added to the denominator to avoid numerical instabilities.

**Theorem 2.** When $\epsilon$-LRP is applied to a network with Relu activation, the attribution of $x_i$ is equivalent to the attribution in Gradient*Input, i.e.,

$$a_i^{\epsilon-LRP} = a_i^{GI},$$

Theorem 2 has previously been proved by Marco [8].

4.1.3 Grad-CAM

Grad-CAM [20] focuses on interpreting the classification module of CNNs and takes the feature maps $z \in \mathbb{R}^{U \times V \times K}$ of the top convolutional layer to calculate the attribution scores. Here $K$ is the number of channels, $U$ and $V$ represent the weight and height of these feature maps. Specifically, Grad-CAM firstly captures the importance of each feature map by conducting global average pooling (GAP) operation to the gradient of the target output neuron $y^c$ w.r.t the feature map. For $k$-th feature map, the importance $\alpha_k^c$ is calculated by

$$\alpha_k^c = \frac{1}{|U| \times |V|} \sum_i \sum_j \frac{\partial y^c}{\partial z_{ij}^c}.$$ 

where $z_{ij}^c$ is the intermediate feature at $(i, j)$ location at $k$-th feature map. Then Grad-CAM can approximately decompose $y^c$ as a weighted combination between the importance weights and these feature maps, i.e.,

$$y^c \approx \sum_k \alpha_k^c \sum_i \sum_j z_{ij}^k. \tag{4}$$

We obtain that Grad-CAM assigns $a_k^c$ contribution to the $k$-th feature map, where $a_k^c$ is expressed as:

$$a_k^c = \sum_i \sum_j z_{ij}^k. \tag{5}$$

To investigate the contributions of the features at different locations, the right side of the Equation 4 can be rewritten as $\sum_i \sum_j (\sum_k \alpha_k^c z_{ij}^k)$. Then Grad-CAM can correspondingly assign $a_{ij}$ contribution to the feature at $(i, j)$ location, where $a_{ij}$ is expressed as:

$$a_{ij} = \sum_k \alpha_k^c z_{ij}^k.$$ 

Define $F^k = \frac{1}{|U| \times |V|} \sum_i \sum_j z_{ij}^k$ to be the $k$-th GAP feature map, and the model $y^c = f(x)$ can be expressed as a function of GAP feature $F^k$, i.e., $y^c = h(F^1, \ldots, F^K)$. Then we have Theorem 3.

**Theorem 3.** The Eq. 5 in Grad-CAM can be reformulated as a first-order Taylor attribution of function $h(F^1, \ldots, F^K)$ w.r.t the baseline point $\tilde{F} = 0$,

$$a_k^{GCA\!M} = T_k^\alpha(h).$$

Specifically, in Grad-CAM, the attribution of $k$-th feature map (Eq. 5) is reformulated as $a_k = \frac{\partial h(F^1, \ldots, F^K)}{\partial F_k} F^k$.

4.1.4 Occlusion-1

Occlusion-1 [10] calculates how much the prediction changes induced by occluding feature $x_i$ with a zero baseline. The new occluded input is written as $\tilde{x}|_{x_i=0}$. Then the attribution of feature $x_i$ is defined as the difference of the output, $a_i = f(\tilde{x}) - f(x|_{x_i=0})$.

**Theorem 4.** The attribution of $x_i$ in Occlusion-1 can be reformulated as the sum of first-order and high-order independent terms of $x_i$ at baseline point $\tilde{x} = x|_{x_i=0}$,

$$a_i^{Oc1} = T_i^\alpha + T_i^{\alpha d}.$$ 

The attribution of $x_i$ in Occlusion-1 is $f_{x_i} \Delta_i + \frac{1}{2} f_{x_i x_i} \Delta_i^2$ in the second-order Taylor attribution.
4.1.5 Occlusion-patch

The attribution in Occlusion-patch [13] is similar to Occlusion-1 but conducted on a patch level. It constructs a zero patch baseline \( x|_{p_j=0} \) by occluding an image patch \( p_j \), and defines the output difference \( f(x) - f(x|_{p_j=0}) \) as the attribution of features in \( p_j \).

**Theorem 5.** The attribution of \( x_i \in p_j \) in Occlusion-patch can be reformulated as the sum of first-order, high-order independent terms of features in patch \( p_j \), and all high-order interactive terms involving the features in patch \( p_j \),

\[
a_i^{OCP} = T_i^\alpha + T_i^\gamma + T_i^\rho = \sum_{x_i \in p_j} T_i^\alpha + \sum_{x_i \in p_j} T_i^\gamma + \sum_{A \subset p_j} T_i^\rho.
\]

Particularly, \( a_i \) is \( \sum_{i \in p_j} \int_{p_j} \int \frac{\partial f(\hat{x} + \alpha(x - \hat{x}))}{\partial x_i} \text{d}x \text{d}y \). In second-order setting.

4.2 With feature interaction

In this subsection, we study four attribution methods considering feature interactions.

4.2.1 Integrated Gradients

The attribution in Integrated Gradients [5] integrates the gradients along the straight line path from a baseline point \( \hat{x} \) to an input \( x \). The points along the path are denoted as \( x' = \hat{x} + \alpha(x - \hat{x}), \alpha \in [0, 1] \). The attribution of feature \( x_i \) is computed by

\[
a_i = (x_i - \hat{x}_i) \int_0^1 \frac{\partial f(\hat{x} + \alpha(x - \hat{x}))}{\partial x_i} \text{d} \alpha.
\]

**Theorem 6.** The attribution of \( x_i \) in Integrated Gradients can be reformulated as the sum of first-order term of \( x_i \), high-order independent terms of \( x_i \), and an assignment from high-order interactive terms involving \( x_i \) at baseline \( \hat{x} \),

\[
a_i^{IG} = T_i^\alpha + T_i^\gamma + a_i^\alpha(IG),
\]

where \( a_i^\alpha(IG) = \sum K \sum k_i \prod_{j=1}^{K} (K_j^k \prod_{j=1}^{K} k_j^{K_j} C \Delta_j^{K_j}) \) is the assignment, and \( C = \frac{1}{K!} \prod_{j=1}^{K} \frac{1}{k_j!} \) is the Taylor expansion coefficient of \( \Delta_1 \ldots \Delta_K \).

In brief, Integrated Gradients assigns \( \frac{k_i^2}{n} \) proportion of the high-order interactive term \( \Delta_1^k \ldots \Delta_K_k \) to \( x_i \).

**Remark 1.** Give a concrete example. Assume \( g(x) = x_1 + x_2 + x_3 + x_1x_2 + x_1x_2x_3^2 \), and let \( x = [x_1, x_2, x_3]^T \) and \( \hat{x} = 0 \). Obviously, the independent terms \( x_1, x_2, x_3 \) should be clearly assigned to \( x_1, x_2, x_3 \) respectively. With respect to the interactive terms, the assignment of Integrated Gradients is based on the order of features, which allocates \( \frac{k_i^2}{n^2} \) proportion of \( x_1^k \ldots x_n^k \) term to feature \( x_i \). Hence, it assigns \( \frac{1}{n} x_1x_2 + \frac{2}{n} x_1x_2^2x_3 \) to feature \( x_1 \), assigns \( \frac{1}{n} x_1x_2 + \frac{2}{n} x_1x_2x_3^2 + \frac{1}{n} x_1x_2x_3^3 \) to feature \( x_2 \), and assigns \( \frac{1}{n} x_1x_2x_3 \) to feature \( x_3 \).

4.2.2 DeepLIFT Rescale

Similar to e-LRP, DeepLIFT Rescale [14] also computes the relevance scores by relevance propagation in a layer-wise manner. While instead of merely considering the output value, DeepLIFT propagates the output difference between the input \( x \) and the baseline \( \hat{x} \) to the input layer. Specifically,
at l-th layer, it calculates the relevance score of $x_i^{(l)}$ to $x_j^{(l+1)}$, denoted as $a_{ij}^{(l)}$, by

$$a_{ij}^{(l)} = \frac{\tilde{y}_j^{(l)} - \tilde{y}_j^{(l)}}{\sum_{i'} \tilde{z}_j^{(l)} - \sum_{i'} \tilde{z}_j^{(l)}}$$.

where $\tilde{z}_j^{(l)} = w_{ji} x_i^{(l)}$ is the weighted impact of $x_i^{(l)}$ to $x_j^{(l+1)}$, analogously $\tilde{z}_j^{(l)} = w_{ji} x_i^{(l)}$ denotes the weighted impact of the baseline, and $a_{ij}^{(l+1)} = \sum_k a_{jk}^{(l+1)}$ denotes the total relevance score of $x_j^{(l+1)}$.

**Theorem 7.** The attribution of $x_i$ in DeepLIFT Rescale at l-th layer is equivalent to the attribution in Integrated Gradients.

$$a_{i}^{DL}(l) = a_{i}^{IG}(l) = T_i^a + T_i^{3a} + a_i^{3i}(IG).$$

Hence DeepLIFT Rescale can be considered as a layer-wise Integrated Gradients.

### 4.2.3 Shapley value

Shapley value [3, 21] is a classical solution concept in cooperative game theory, which aims to assign an importance score to each player (feature) in a cooperative game (model) involving the coalition of $n$ players (features). According to Shapley value, given a cooperative game $f(X)$, the amount that player $i$ contributes is,

$$a_i = \frac{1}{n} \sum_{k=0}^{n-1} \sum_{S \subseteq X \setminus \{i\} \atop |S| = k} (n-1)^{-1} (f(S \cup \{i\}) - f(S)).$$

Here $X$ is the set of all players, and $S$ traverses all subsets of $X \setminus \{x_i\}$. Eq. 9 can be interpreted as averaging the marginal contribution of $x_i$ to coalition over all possible coalitions (involving $x_i$). When applied to interpreting DNNs, $f(S)$ is often obtained by calculating the output when setting the values of complementary set $S$ as 0.

**Theorem 8.** The attribution of $x_i$ in Shapley value can be reformulated as the sum of first-order term of $x_i$, high-order independent terms of $x_i$, and $\frac{1}{|S|+1}$ proportion of the interactive terms among features in $S \cup \{i\}$.

$$a_i^{Shap} = T_i^a + T_i^{3a} + a_i^{3i} (Shap),$$

$$a_i^{3i} (Shap) = \frac{1}{|S|+1} \sum_{k=0}^{n-1} \sum_{S \subseteq X \setminus \{i\} \atop |S| = k} T_i^{3a}.$$ 

In other words, Shapley value averagely assigns the interactive terms $T_i^{3a} (Shap)$ among features in the set $S \cup \{i\}$, i.e., the proportion of each feature is $\frac{1}{|S|+1}$.

**Remark 2.** For example, let $g(x) = x_1 x_2 + x_1^2 x_2 + x_2 x_2 x_3$.

The interactive terms involving two features are $x_1 x_2$ and $x_1^2 x_2$, respectively. The terms involving three features are $x_1 x_2 x_3^2$. Hence, Shapley value assigns $\frac{1}{2}(x_1 x_2 + x_1^2 x_2)$ to feature $x_1, x_2$, and assigns $\frac{1}{2} x_1^2 x_2 x_3^2$ to feature $x_1, x_2, x_3$.

### 4.3 Separating positive and negative contributions

Shrikumar [14] has shown that the positive and negative impacts may cancel out during the attribution process and hence may provide misleading interpretations. To alleviate such issues, some variants including DeepLIFT RevealCancel [14], Deep Taylor [13], and LRP-αβ [6], have been proposed to treat the positive and negative impacts separately.

The basic idea is to firstly decompose the output difference $\Delta y$ into positive components $\Delta y^+$ arising from positive input differences and negative components $\Delta y^-$ arising from negative input differences. Then $\Delta y^+$ is allocated to the neurons with positive input differences, and $\Delta y^-$ proceeds analogously. We give a unified formulation for the three attribution methods.

Similar to DeepLIFT Rescale, the three attribution methods also proceed in a recursively back-propagation manner. Hence we adopt the same set of symbols as in DeepLIFT Rescale. We focus on the propagation from the target neuron $j$ at $(l + 1)$-layer to the input neurons at $l$-th layer. For sake of simplicity, we omit the superscript of layer index (i.e., $l$ and $l + 1$) and subscript of target neuron index (i.e., $j$). In addition, we rewrite all input neurons as $x$, and denote the target output neuron as $y$. We represent the corresponding weighted impacts $\{z_i = w_i x_i\}$ as $z$. Hence we have $y = \sigma(z + b)$. These three methods firstly decomposes the input differences $\Delta z$ into positive and negative parts, i.e.,

$$\Delta z = \Delta z^+ + \Delta z^-,$$

where $\Delta z^+ = \max(\Delta z, 0)$ and $\Delta z^- = \min(\Delta z, 0)$. Specifically, for each input neuron $i$, we define $\Delta z_i^+ = \max(\Delta z_i, 0), \Delta z_i^- = \min(\Delta z_i, 0)$.

Moreover, we denote $A^+ = \{i|w_i \Delta x_i \geq 0\}$ and $A^- = \{i|w_i \Delta x_i < 0\}$ as the feature subset with positive $\Delta z_i$ and the feature subset with negative $\Delta z_i$, respectively.

These attribution methods decompose the output difference $\Delta y$ into the positive component $\Delta y^+$ and the negative component $\Delta y^-$, which satisfies $\Delta y^+ - \Delta y^- = \Delta y$. For the features in $A^+$, the positive attribution $a_i^+$ of feature $x_i$ from $j$-th neuron at $(l + 1)$-th layer is obtained by,

$$a_i^+ = \frac{\Delta z_i^+}{\Delta z_i^+} \frac{\Delta y^+}{\Delta y} a_j^{(l+1)} = \frac{(w_i \Delta x_i)^+}{\sum_{i'} (w_i \Delta x_i')^+} \frac{\Delta y^+}{\Delta y} a_j^{(l+1)}.$$ 

Similarly, for the features in $A^-$, the negative attribution $a_i^-$ of $x_i$ from $j$-th neuron at $(l + 1)$-th layer is obtained by,

$$a_i^- = \frac{\Delta z_i^-}{\Delta z_i^-} \frac{\Delta y^-}{\Delta y} a_j^{(l+1)} = \frac{(w_i \Delta x_i)^-}{\sum_{i'} (w_i \Delta x_i')^-} \frac{\Delta y^-}{\Delta y} a_j^{(l+1)}.$$ 

If $w_i \Delta x_i \geq 0$, then $(w_i \Delta x_i) = 0$ and $a_i^+ = 0$. Conversely, if $w_i \Delta x_i \leq 0$, then $(w_i \Delta x_i) = 0$ and $a_i^- = 0$.

How to separate $\Delta y^+$ and $\Delta y^-$ is essential for the attribution. Different definitions of $\Delta y^+$ and $\Delta y^-$ are adopted in the three attribution methods, which are summarized in Table 3 and will be further introduced in the following sections.
DeepLIFT RevealCancel method [14] introduces $\Delta y^+$ and $\Delta y^-$ to represent the positive and negative components of output difference $\Delta y$, which are defined by considering the overall marginal impact of $\Delta z^+$ and $\Delta z^-:
\[
\Delta y^+ = \frac{1}{2}(\sigma(\tilde{z} + \Delta z^+) - \sigma(\tilde{z})) + \frac{1}{2}(\sigma(\tilde{z} + \Delta z^+ + \Delta z^-) - \sigma(\tilde{z} + \Delta z^-))
\]
\[
\Delta y^- = \frac{1}{2}(\sigma(\tilde{z} + \Delta z^- - \sigma(\tilde{z})) + \frac{1}{2}(\sigma(\tilde{z} + \Delta z^+ + \Delta z^-) - \sigma(\tilde{z} + \Delta z^+)).
\]

Here $\Delta y^+$ and $\Delta y^-$ can be thought of the Shapely values of $\Delta z^+$ and $\Delta z^-$ contributing to $\Delta y$.

The positive and negative attribution in DeepLIFT RevealCancel follow Eq. 10 and Eq. 11. It’s easy to demonstrate that DeepLIFT RevealCancel satisfies the completeness axiom at each layer, i.e., $\sum_i (a_i^+ + a_i^-) = a_i^{(l+1)}$.

**Theorem 9.** For the features $x_i$ in the subset $A^+$, the attribution in DeepLIFT RevealCancel can be reformulated as follows:
\[
a_i^{DL+} = T_i^a + T_i^{a\gamma} + a_i^{\gamma} (DL+).
\]
\[
a_i^{\gamma} (DL+) = a_i^{\gamma} (IG) + \frac{1}{2} a_i^{\gamma} (IG),
\]
where $a_i^{\gamma} (IG)$ represents the assignment from the interactive terms between $x_i$ and other features in $A^+$. Similarly, $a_i^{\gamma} (IG)$ represents the assignment from the interactive terms between $x_i$ and all features in subset $A^-$. Such assignment follows the same rule as in Integrated Gradients.

Similarly, for the features $x_i$ in the subset $A^-$, the conclusion is derived analogously:
\[
a_i^{DL-} = T_i^a + T_i^{a\gamma} + a_i^{\gamma} (DL-).
\]
\[
a_i^{\gamma} (DL-) = a_i^{\gamma} (IG) + \frac{1}{2} a_i^{\gamma} (IG),
\]
The main difference between DeepLIFT RevealCancel and DeepLIFT Rescale is how to attribute the interactive terms $T_{\gamma A^+, A^-}$ among features in $A^+$ and $A^-$. DeepLIFT Rescale never considers whether $x_i$ is in subset $A^+$ or $A^-$, while in DeepLIFT RevealCancel, the features in $A^+$ and $A^-$ are not treated in a same manner.

**Remark 3.** For example, assume the Taylor expansion at l-th layer is $g^l(x) = x_1 x_2^2 + x_1 x_2 x_3 + x_1 x_2 x_3$. We consider $A^+ = \{x_1, x_3\}$, and $A^- = \{x_2\}$. DeepLIFT RevealCancel firstly averagely assign $T_i^{\gamma A^+, A^-}$ (i.e., $x_1 x_2^2 + x_1^3 x_2 + x_1 x_2 x_3^2$) to positive and negative subset $A^+$ and $A^-$, i.e.,
\[
a_{A^+} = \frac{1}{2}(x_1 x_2^2 + x_1^3 x_2 + x_1 x_2 x_3^2 + x_1^3),
\]
\[
a_{A^-} = \frac{1}{2}(x_1 x_2^2 + x_1^3 x_2 + x_1 x_2 x_3^2).
\]
Then in $A^+$ and $A^-$, features are again assigned according the same rule with Rescale, such as $x_1$ and $x_3$. It assigns $\frac{1}{2}(x_1 x_2^2 + x_1^3 x_2 + x_1 x_2 x_3^2 + x_1^3)$ to $x_1$, and $a_3 = \frac{1}{2}(x_1 x_2^2 + x_1^3 x_2 + x_1 x_2 x_3^2 + x_1^3)$ to $x_3$.

4.3.2 Deep Taylor

The attribution in Deep Taylor [15] is obtained by measuring the positive weighted impacts of the input neuron $i$ to the target output neuron $j$ and normalizing by the total weighted impacts to neuron $j$. The process can be formalized as:
\[
a_i^{DTD} = \frac{(w_i \Delta x_i)^+}{\sum_j (w_i \Delta x_j)^+} a_j^{(l+1)}
\]
Comparing with the Eq. 10 and Eq. 11, we consider that DTD is a special case of Eq. 10 and Eq. 11. Where $\Delta y^+$ and $\Delta y^-$ are defined as:
\[
\Delta y^+ = \Delta y, \quad \Delta y^- = 0.
\]
That is, the output differences $\Delta y$ resulted by both $\Delta z^+$ and $\Delta z^-$, are all assigned to the features with positive $\Delta z_i$. Zero of $\Delta y$ are allocated to the features with negative $\Delta z_i$. Obviously, such assignment is unreasonable, which is further proved in Theorem 10.

**Theorem 10.** The attribution in Deep Taylor can be reformulated as follows:
\[
a_i^{DT} = T_i^a + T_i^{a\gamma} + a_i^{\gamma} (IG) + \frac{\Delta z^+}{\Delta z^+} T_{A^-},
\]
where $T_{A^-} = T_{A^-} + T_{A^-} + T_{A^-} + T_{A^-}$ denotes the sum of the first-order, high-order independent and high-order interactive terms involving all features in $A^-$. According to Theorem 10, Deep Taylor wrongly assigns the extra contributions (i.e., $T_{A^-}$) from the features in subset $A^-$ to the features in subset $A^+$.

4.3.3 LRP-$\alpha\beta$

The attribution of LRP-$\alpha\beta$ is calculated by,
\[
a_i = (\alpha \cdot \frac{(w_i \Delta x_i)^+}{\sum_j (w_i \Delta x_j)^+} - \beta \cdot \frac{(w_i \Delta x_i)^-}{\sum_j (w_i \Delta x_j)^-}) a_j^{(l+1)},
\]
where $\alpha$ and $\beta$ satisfy $\alpha - \beta = 1$. Similar to Deep Taylor, we can consider that LRP-$\alpha\beta$ is a special case of Eq. 10 and Eq. 11. Where $\Delta y^+$ and $\Delta y^-$ are defined as:
\[
\Delta y^+ = \alpha \Delta y, \quad \Delta y^- = -\beta \Delta y.
\]
That is, the output differences $\Delta y$ is firstly decomposed as $\Delta y = \alpha \Delta y - \beta \Delta y$. Then $\alpha \Delta y$ is assigned to the features with positive $\Delta z_i$, while $-\beta \Delta y$ is allocated to the features with negative $\Delta z_i$. 

| Methods       | Formulation                                                                 | Remarks            |
|---------------|----------------------------------------------------------------------------|--------------------|
| DeepLIFT Reveal | $a_i^+ = \frac{\Delta z_i^+}{\Delta z^+}$, $a_i^- = \frac{\Delta z_i^-}{\Delta z^-}$ | $\Delta y^+ = $ Eq. [12] $\Delta y^- = $ Eq. [13] |
| Deep Taylor   | $a_i^+ = \frac{\Delta z_i^+}{\Delta z^+}$, $a_i^- = \frac{\Delta z_i^-}{\Delta z^-}$ | $\Delta y^+ = $ $\Delta y$, $\Delta y^- = 0.$ |
| LRP-$\alpha\beta$ | $a_i^+ = \frac{\Delta z_i^+}{\Delta z^+}$, $a_i^- = \frac{\Delta z_i^-}{\Delta z^-}$ | $\Delta y^+ = \alpha \Delta y$, $\Delta y^- = -\beta \Delta y.$ |
**Theorem 11.** For the features \( x_i \) in \( A^+ \), the attribution in LRP-\( \alpha \beta \) can be reformulated as follows:

\[
a^{LRP-\alpha \beta +}_i = \alpha(T^{\alpha}_i + T^{\gamma}_i + a^\gamma_i(IG) + \frac{\Delta z^+_i}{\Delta z^+}, T_{A^-}^+),
\]

where \( T_{A^-} = T_{A^-}^0 + T_{A^-}^{\gamma} + T_{A^-}^\alpha + T_{A^-}^{\alpha, A^-} \). For the features \( x_i \) in \( A^- \), the conclusion is derived analogously:

\[
a^{LRP-\alpha \beta -}_i = \beta(T^{\alpha}_i + T^{\gamma}_i + a^\gamma_i(IG) + \frac{\Delta z^-_i}{\Delta z^-}, T_{A^+}^-),
\]

where \( T_{A^+} = T_{A^+}^0 + T_{A^+}^{\gamma} + T_{A^+}^\alpha + T_{A^+}^{\alpha, A^+} \).

Similar to Deep Taylor, such decomposition is not thoughtful, leading to unreasonable attributions.

### 4.4 Expected Attribution

The selection of baseline point is essential to the performance of attribution methods, as discussed in section 3.4. When interpreting the decision of image classification, the constant vector baseline (\( \tilde{x} = c \)) usually introduces biases into the attribution.

An effective approach to neutralize these biases is expected attribution, which extends existing attribution methods by averaging over multiple baselines. It reduces the probability that the attribution is dominated by a specific baseline. Denote the attribution of feature \( x_i \) by basic method as \( a^{basic}_i(\tilde{x}) \). Specifically, expected attribution samples baseline points from a prior distribution \( p_D(\tilde{x}) \), the corresponding attribution \( a^{exp}_i \) is then computed by integrating \( a^{basic}_i \) along the baseline distribution,

\[
a^{exp}_i = \int_{\tilde{x}} p_D(\tilde{x}) a^{basic}_i(\tilde{x}) d\tilde{x} . \tag{15}
\]

A common choice for prior distribution is an independent Gaussian distribution around \( x \). That is, \( \tilde{x} \sim N(x, \sigma^2) \).

In this subsection, we investigate three effective expected attribution methods, which extends Integrated Gradients, DeepLIFT Rescale, and Shapley value, respectively.

#### 4.4.1 Expected Gradients

Expected Gradients [19] is an extension of Integrated Gradients, where the attribution is

\[
a_i = \int_{\tilde{x}} p_D(\tilde{x}) a^{IG}_i(\tilde{x}) d\tilde{x} .
\]

**Theorem 12.** The attribution of feature \( x_i \) in Expected Gradients can be reformulated as:

\[
a^{EG}_i = \int_{\tilde{x}} p_D(\tilde{x})(T^{\alpha}_i + T^{\gamma}_i + a^\gamma_i(IG)) d\tilde{x},
\]

Here, \( T^{\alpha}_i, T^{\gamma}_i \), and \( a^\gamma_i(IG) \) depends on the baseline point \( \tilde{x} \).

#### 4.4.2 Expected DeepLIFT

Expected DeepLIFT is an extension of DeepLIFT Rescale, where the attribution is expressed as:

\[
a_i = \int_{\tilde{x}} p_D(\tilde{x}) a^{DL}_i(\tilde{x}) d\tilde{x} .
\]

**Theorem 13.** The attribution of feature \( x_i \) at \( l \)-th layer in Expected DeepLIFT can be reformulated as:

\[
a^{ED}_i(l) = \int_{\tilde{x}} p_D(\tilde{x}) a^{IG}_i(l)(\tilde{x}) d\tilde{x} = \int_{\tilde{x}} p_D(\tilde{x})(T^{\alpha}_i + T^{\gamma}_i + a^\gamma_i(IG)) d\tilde{x} .
\]

Here, \( T^{\alpha}_i, T^{\gamma}_i \), and \( a^\gamma_i(IG) \) at \( l \)-th layer depends on \( \tilde{x} \).

#### 4.4.3 Deep Shapley

Deep Shapley [3] is an extension of layer-wise Shapley value, where the attribution is,

\[
a^{DShap}_i = \int_{\tilde{x}} p_D(\tilde{x}) a^{Shap}_i(\tilde{x}) d\tilde{x} .
\]

**Theorem 14.** The attribution in Deep Shapley at each layer is an approximation of Shapley value. Hence the attribution of feature \( x_i \) at \( l \)-th layer can be reformulated as:

\[
a^{DShap}_i(l) \approx \int_{\tilde{x}} p_D(\tilde{x}) a^{Shap}_i(l)(\tilde{x}) d\tilde{x} = \int_{\tilde{x}} p_D(\tilde{x})(T^{\alpha}_i + T^{\gamma}_i + a^\gamma_i(Shap)) d\tilde{x} .
\]

Here, \( T^{\alpha}_i, T^{\gamma}_i \), and \( a^\gamma_i(Shap) \) at \( l \)-th layer depends on \( \tilde{x} \).

## 5 Revisiting Attribution Methods

In this section, we firstly revisit the aforementioned fourteen attribution methods in a systematic and theoretical way, according to the unified Taylor reformulations. Then we establish and advocate three principles for a good Taylor attribution, which are low approximation error, correct Taylor contribution assignment, and unbiased baseline selection.

### 5.1 Theoretical Analysis of the Attribution Methods

The unified Taylor reformulations enable us to reveal rationales, measure fidelity, and point out limitations for the attribution methods in a systematic and theoretical way.

#### Without feature interaction

Firstly, we find most attribution methods without feature interaction, including Gradient*Input, \( \epsilon \)-LRP, Occlusion-1, and Occlusion-patch, have a large Taylor approximation error \( \epsilon \) and thus achieve low fidelity. This is because all of them fail to completely reflect the high-order Taylor interactive terms, while such complex interactions among features always contains critical information for prediction in DNNs. For example, it’s infeasible to classify objects by considering each pixel in isolation. The interactions of pixels and super-pixels are very common and essential for successful recognition.

Let’s take a closer look at the above four attribution methods. Specifically, Theorem [1] and [2] show that Gradient*Input and \( \epsilon \)-LRP are both first-order Taylor attribution, which only takes the first-order terms into consideration. Occlusion-1 characterizes the high-order independent effects (Theorem [3]), however it fails to attribute the interactive terms. Theorem [5] shows Occlusion-patch considers both independent and interactive effects of the features in the patch. However, it assigns the same contribution score to all
features in the patch, which fails to provide fine-grained attributions. Moreover, the interactive effects among different patches are neglected in Occlusion-path.

There is one exception, Grad-CAM, which has a low Taylor approximation error and achieves significantly higher fidelity. Although Theorem 3 demonstrates that Grad-CAM can be reformulated as a first-order Taylor approximation, it’s worthy noting that different from the other four methods, Grad-CAM considers the model as a function of GAP top convolutional features (i.e., $[F^1, \ldots, F^K]$), instead of a function of feature-level/pixel-level input. That is, Grad-CAM actually interprets the classifier module of CNNs, instead of the whole model. This module is a Multilayer perceptron (MLP) structure, and is much easier to be approximated by the first-order Taylor expansion. Hence, Grad-CAM usually produces good interpretations results. However, Grad-CAM is incapable of interpreting the feature extraction module, and the pixel-level explanation can only be implemented by resize operation. Therefore, Grad-CAM fails to produce fine-grained explanation and can’t be applied to the models without localization abilities structures (e.g., convolutional layers).

**With feature interaction** From the Taylor reformulations, we can see the three attribution methods with feature interaction, which are Integrated Gradients, DeepLIFT Rescale, and Shapley value, attribute the high-order Taylor terms completely and reasonably. So they have a zero Taylor approximation error $c$.

In summary, the attributions of the three methods can be uniformly written as the following form: $a_i = T_i^a + T_i^{ad} + a_i^u$, where $(T_i^a + T_i^{ad})$ means the Taylor first-order terms and high-order independent terms of feature $x_i$ are assigned to $x_i$. While $a_i^u$ indicates that part of Taylor interactive terms $T_i^x$ among features in subset $A$ (including feature $x_i$) are allocated to $x_i$. Furthermore, in these three methods, $T_i^x$ are assigned and only assigned to the features in the subset $A$, i.e., $\sum_{i \in A} a_i^x = T_i^x$. This theoretical finding may provide an insight into why the three methods can well identify the important features to prediction.

Next we will conduct a further comparison among the three methods. Firstly, the main difference between Integrated Gradients and Shapley value lies on how to decide the assignment $a_i^u$ from the interactive terms. Theorem 6 shows that Integrated Gradients assigns $T_i^x$ according to the order of features, while Theorem 8 demonstrates that Shapley value assigns $T_i^x$ based on the number of features in the coalition. For example, let $g(x) = x_1^2 x_2^3 + x_1^2 x_2 + x_1^2 x_2 x_3^3$. Then, the $a_i^x$ in Integrated Gradients is,

$$a_i^x (IG) = \frac{1}{3} x_1 x_2^3 + \frac{3}{4} x_1^2 x_2 + \frac{3}{4} x_1^2 x_2 x_3^3.$$  

Correspondingly, the $a_i^u$ in Shapley value is,

$$a_i^u (Shap) = \frac{1}{4} (x_1 x_2^2 + x_1^2 x_2) + \frac{1}{4} (a_2^x x_2 x_3^3).$$

We give a concrete example to show their differences. Assume $x_1 = 10, x_2 = 2, x_3 = 1$, the attributions of $x_1$ and $x_2$ is Integrated Gradients are $a_1^x (IG) = 1593.3, a_2^x (IG) = 566.7$, respectively. While in Shapley value, we have $a_1^x (Shap) = 1086.7, a_2^x (Shap) = 1086.7$. It can be observed that when allocating contributions in coalition, Integrated Gradients would consider some players may contribute more to the coalition than others (e.g., $x_1 > x_2$), and deserves higher payoff. While Shapley value considers the players in the coalition have equal contributions, and would prefer assigning equal payoffs. Both kinds of allocation are reasonable in logic. Which one to choose should depend on which assumption is satisfied in a special case. For example, when $x_1, x_2, x_3$ separately denotes the investment amount from three cooperators and $g(x)$ represents the final benefits. Then the player with higher pay deserves higher return, where Integrated Gradients is more applicable. When $x_1, x_2, x_3$ denotes the adjacent pixels in a patch and $g(x)$ is an image classification model, it’s biased to say higher pixel values would contribute more to the discriminative ability. We usually believe that the adjacent pixels in the patch have similar attribution, where Shapley value is more suitable.

Secondly, DeepLIFT Rescale is a layer-wise version of Integrated Gradients, as indicated in Theorem 7. Specifically, denote the the $j$-th upper neuron $x_{l+1}^j$ at $(l+1)$-th layer as function of the bottom neurons $x^j$, i.e., $x_{l+1}^j = f^j_{l+1}(x^j)$. Consider the sub-model $f^j_l(x^j)$, the attribution of DeepLIFT Rescale follows the same rule with Integrated Gradients. That is, $a_{l}^{DL} = T_{l}^a(f^j_l) + T_{l}^{ad}(f^j_l) + a_{l}^u(f^j_l)$. Hence, the attribution explanations of DeepLIFT Rescale would be similar to Integrated Gradients when the number of layers is small.

Based on the analysis, we point out limitations of the three methods. On one hand, consider interpreting image classification prediction, the attribution performance of Integrated Gradients and DeepLIFT Rescale highly depends on baseline. The explanation would be biased if there is a large difference among $\Delta i$ of different features. On the other hand, Shapley value has an exponential complexity $O(2^n)$, where $n$ denotes the dimension of features. Therefore, Shapley is usually applied in low-dimensional tabular data (citation) and not scalable for high-dimensional cases, such as image.

**Expected Attribution** Expected Attribution approach relieved the problem induced by biased baseline by averaging among multiple baselines, as analyzed in section 3.4. It reduces the probability that the attribution is dominated by a specific baseline. Hence, the saliency maps produced by Expected Attribution are usually evenly distributed with less noises, sharper, and can clearly display the shapes and boarders of the objects.

**Separating positive and negative contributions** DeepLIFT RevealCancel, Deep Taylor and LRP-$\alpha$ try to disentangle the contributions caused by positive and negative components. Different separation strategies are adopted in the three methods.

The positive and negative separation of DeepLIFT RevealCancel follows the Shapley value. Specifically, $\Delta y^+$ is an average of the marginal contribution (to coalition) over all coalitions, which is caused by positive components. $\Delta y^-$ is derived analogously. Theorem 9 indicates that DeepLIFT RevealCancel has a correct Taylor contribution assignment. The main difference between Rescale and RevealCancel is how to assign $T_{i, A^+}^x$ and $T_{i, A^+}^x$. As shown in Eq. 14 due to disentangling positive and negative contributions, the assignment of
RevealCancel is closer to the assignment of Shapley value. However, Deep Taylor fails to correctly assign the Taylor interactive terms. Specifically, $\Delta y^+ = \Delta y$ and $\Delta y^- = 0$ in Deep Taylor. It can be seen that $\Delta y$, which is caused by both positive and negative components, is all allocated to positive components. As shown in Theorem 10, Deep Taylor would wrongly assign the extra contributions $T_{A^-}$ from the negative feature subset to the features in positive subset. LRP-$\alpha\beta$ has the similar drawback. Therefore, we believe the two methods are improper for attribution.

### 5.2 Three Principles of Attribution

Based on the reformulations and analysis, we find a good Taylor attribution depends on three key factors: i) the Taylor approximation error $\epsilon(\hat{x}, K)$; ii) whether the Taylor terms in $g_K(x, \Delta)$ are assigned correctly; iii) the baseline point $\hat{x}$. Accordingly, we establish three principles of a good attribution and advocate the principles should be followed by other attribution methods.

**First principle:** After Taylor reformulation, an attribution method should have a low approximation error $\epsilon(\hat{x}, K)$, $\forall \hat{x}$. This principle is similar to the completeness axiom.

**Second principle:** After Taylor reformulation, an attribution method should have a correct Taylor contribution assignment.

**Third principle:** After Taylor reformulation, an attribution method should choose an unbiased baseline.

Table 4 presents a summary of the principles followed by the attribution methods, in which Occlusion-1 and Occlusion-p partially follow the second principle because they partially attribute high-order terms, as discussed above. In addition, GradCAM is a first-order Taylor attribution with a small Taylor approximation error, and don’t involve the high-order terms. Hence, we consider GradCAM satisfies all three principles.

---

**Table 4:** A summary of the principles followed by the attribution methods.

| Categorization | Method       | First | Second | Third |
|----------------|--------------|-------|--------|-------|
| Without Interaction | GI           |       |        |       |
|                   | LRP          | ✓     | ✓      | ✓     |
|                   | GCAM         | ✓     | ✓      | ✓     |
|                   | Occ-1&p      | partially |       |       |
| With Interaction  | Integrated   | ✓     | ✓      |       |
|                   | DeepLIFT     | ✓     | ✓      |       |
|                   | Shapley      | ✓     | ✓      | ✓     |
| Separating + and - | DeepLIFT+−   | ✓     | ✓      |       |
|                   | Deep Taylor  | ✓     |        |       |
|                   | LRP-$\alpha\beta$ | ✓     |        |       |
| Expected Attribution | E-Integrated | ✓     | ✓      | ✓     |
|                   | E-DeepLIFT   | ✓     | ✓      | ✓     |
|                   | Dshap        | ✓     | ✓      | ✓     |

Based on Table 4 we will recommend the attribution methods which satisfies the above three principles. We recommend GradCAM for a coarse attribution in CNN architectures, and recommend Shapley value for attribution in a low-dimensional setting. While Expected Gradients, Expected DeepLift, and Dshap are more generalizable in high-dimensional setting and other architectures.

### 6 Conclusion and Future Work

In this work, we propose a general Taylor attribution framework to models the attribution problem as how to decide individual payoffs in a coalition. In addition, we theoretically reformulate fourteen mainstream attribution methods into the Taylor attribution framework. Based on reformulations, we systematically analyze the attribution methods in terms of their rationale, fidelity, and limitation. Based on the reformulation and analysis, we establish three principles for a good attribution. In the future work, we will consider more how to fully utilize the interaction terms in attribution explanations.
Proof of Theorem 5

Proof. Let \( F^k = \frac{1}{|U| \times |V|} \sum_{i,j} z^k_{ij} \) to be the \( k \)-th GAP feature map, then the model \( f(x) \) can be represented as a function of GAP feature \( F^k \), i.e., \( f(x) = h(F^1, \ldots, F^K) \). Then we conduct the Taylor expansion of function \( h(F^1, \ldots, F^K) \) at \( F = [F^1, \ldots, F^K] \) point w.r.t \( F = 0 \) as follows:

\[
y^c = h(F) = h(0) + \sum_k \frac{\partial y^c}{\partial F^k} F^k + \epsilon_1(0),
\]

where \( \epsilon_1(0) \) is the first-order expansion error of function \( h \) at \( F \). We compute the gradient of \( y^c \) w.r.t the \( k \)-th GAP feature \( F^k \). According to the chain rule, 

\[
\frac{\partial y^c}{\partial F^k} = \frac{\partial y^c}{\partial \hat{a}} \frac{\partial \hat{a}}{\partial F^k}.
\]

Therefore,

\[
\frac{\partial y^c}{\partial F^k} = \frac{\partial y^c}{\partial \hat{a}} \frac{\partial \hat{a}}{\partial F^k} (\{U\} \times \{V\}) \sum_k \frac{\partial y^c}{\partial z^k_{ij}} (16)
\]

Summing both sides of Eq. [16] over all locations \((i, j)\),

\[
\sum_i \sum_j \frac{\partial y^c}{\partial F^k} = (\{U\} \times \{V\}) \sum_i \sum_j \frac{\partial y^c}{\partial z^k_{ij}}
\]

We can obtain \( \frac{\partial y^c}{\partial F^k} = \sum_i \sum_j \frac{\partial y^c}{\partial z^k_{ij}} \). Then the first-order expansion can be expressed as,

\[
T^\alpha(h) = \sum_k \sum_i \sum_j \frac{\partial y^c}{\partial F^k} F^k = \sum_k \alpha_k \sum_i \sum_j z^k_{ij}.
\]

Then the conclusion holds. 

Proof of Theorem 4

Proof. The attribution of feature \( x_i \) is calculated by \( a_i = f(x) - f(x_{x_i} = 0) \). Here \( x_{x_i} = 0 \) denotes the baseline point that omits feature \( x_i \) of \( x \) while remains other features unchanged, and \( f(x_{x_i} = 0) \) denotes the model output of the baseline. We firstly conduct Taylor expansion at point \( x \) w.r.t baseline point \( x_{x_i} = 0 \). It’s observed that \( \Delta_i = x_i \) while for \( \forall j \neq i, \Delta_j = x_j - x_j = 0 \). By substituting \( \Delta_i = x_i \) into Taylor expansion, we have, for \( \forall j \neq i, \) the first-order term \( T^\alpha \) and all high-order independent terms of other features \( T^\beta \) feature equal to 0. Moreover, all high-order interactive terms \( T^\tau \) also equals to 0. That is,

\[
a_i = f(x) - f(x_{x_i} = 0) = T_i^\alpha + T_{i}^{\beta} + \sum_{j \neq i} (T_j^\alpha + T_j^{\beta} + T_j^{\tau}) = T_i^\alpha + T_{i}^{\beta}.
\]

The attribution of feature \( x_i \) in Occlusion-1 can be reformulated as the sum of first-order and high-order independent terms of \( x_i \) w.r.t baseline point \( x_i = 0 \).

Proof of Theorem 3

Proof. The attribution of feature \( x_i \) in patch \( p_j \) is calculated by \( a_i = f(x) - f(x|p_j=0) \). Here \( x|p_j=0 \) denotes the baseline...
point that occludes all features in patch \( p_j \) of \( x \) while remains other patches unchanged, and \( f(x|p_j=0) \) denotes the model output of the baseline. Similar to the derivation of Occlusion-1, we conduct Taylor expansion at point \( x \) w.r.t baseline point \( x|p_j=0 \). We can see that for all other patches \( \forall k \neq j, \Delta p_k = 0 \). By substituting \( \Delta \) into Taylor expansion, we have, for \( \forall k \neq j \), all first-order terms of features in the patch \( p_k \) equal to 0. That is, \( \sum_{x_m \in p_k} T^m = 0 \). Similarly, all high-order independent terms of features in the patch \( p_k \) equal to 0. i.e., \( \sum_{x_m \in p_k} T^m = 0 \). Moreover, interactive terms among these patches equal to 0. In addition, interactive terms between patch \( p_j \) and other patches also equal to 0. That is, \( \sum_{B \subseteq X \setminus p_j} T^B = 0 \) and \( \sum_{C \subseteq X \setminus p_j} T^C = 0 \). Here \( C \) denotes a feature subset of \( X \) which not only involves features in patch \( p_j \), but also involves features in other patches, e.g., patch \( p_k \).

Therefore,

\[
\begin{align*}
  a_i &= f(x) - f(x|p_j=0) \\
  &= \sum_{i \in p_j} (T_i + T_i^0) + \sum_{k \neq j} \sum_{x_m \in p_k} (T^m + T^m_i) \\
  &+ \sum_{A \subseteq p_j \setminus B} T^A + \sum_{B \subseteq X \setminus p_j} T^B + \sum_{C \subseteq X \setminus p_j} T^C \\
  &= \sum_{i \in p_j} (T_i + T_i^0) + \sum_{A \subseteq p_j} T^A,
\end{align*}
\]

where \( A \) goes through all subsets of feature set \( p_j \). \( T_i^0 \) involves the overall high-order interactive terms among the features in subset \( A \).  

A.2 With feature interaction

Proof of Theorem [6]

Proof. Integrated Gradients [5] integrates the gradient over the straight line path from the selected baseline \( \tilde{x} \) to input \( x \). \( a_i \) could be formulated as

\[
a_i = (x_i - \tilde{x}_i) \int_0^1 \frac{\partial f(\tilde{x} + \alpha(x - \tilde{x}))}{\partial x_i} \, d\alpha,
\]

where \( \sum_i a_i = f(x) - f(\tilde{x}) \). To fit the predefined Taylor expansion form, we rewrite the integral as starting from input \( x \) to baseline \( \tilde{x} \),

\[
a_i = (x_i - \tilde{x}_i) \int_0^1 \frac{\partial f(x + \alpha(\tilde{x} - x))}{\partial x_i} \, d\alpha
\]

\[
= -\Delta_i \int_0^1 \frac{\partial f(x + \alpha(\tilde{x} - x))}{\partial x_i} \, d\alpha
\]

Note that \( \int_0^1 \frac{\partial f(x + \alpha(\tilde{x} - x))}{\partial x_i} \, d\alpha \) is a function of \( \Delta \), denoted as \( g(\Delta) \). We prove the theorem by conducting Taylor expansion on function \( g(\Delta) \) at 0 point. Firstly, the value of \( g(\Delta) \) at 0 point is:

\[
g(0) = \int_0^1 \frac{\partial f(x)}{\partial x_i} \, d\alpha = \frac{\partial f(x)}{\partial x_i}.
\]

Then the first order partial derivative of \( g(\Delta) \) with respect to \( \Delta \) at 0 point is calculated as:

\[
g_{\Delta_i}(0) = \left. \frac{\partial^1 f(x + \alpha \Delta)}{\partial \Delta_i} \right|_{\Delta=0} = \int_0^1 \frac{\partial f(x + \alpha \Delta)}{\partial \Delta} \, d\alpha = \frac{\partial f(x)}{\partial \Delta_i} \int_0^1 \theta \, d\alpha
\]

Similarly, the first order partial derivative with respect to \( \Delta_j \) at 0 point is:

\[
g_{\Delta_j}(0) = \frac{1}{2} \frac{\partial f(x)}{\partial \Delta_i \partial \Delta_j}.
\]

So the first order Taylor expansion of \( g \) at 0 becomes,

\[
g(\Delta) = g(0) + \sum_k g_{\Delta_k}(0) \Delta_k + \epsilon
\]

where \( \epsilon \) denotes expansion error. Correspondingly, as \( a_i = -\Delta_i g(\Delta) \), we further obtain (assume \( f_{x_i} = f_{x_i, x_j} \)),

\[
a_i = -\frac{\partial f(x)}{\partial x_i} \Delta_i + \frac{1}{2} \frac{\partial^2 f(x)}{\partial x_i \partial x_k} \Delta_k + \frac{1}{2} \sum_{j \neq i} \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \Delta_j + \epsilon
\]

Hence, in second-order setting, Integrated Gradients assigns first-order term \( T_i \), second-order independent term \( T^0_i \) and half of all second-order interactive terms associating with feature subset \( \{x_i, x_j\} \), i.e., \( \sum_{i, j} T^i_{x_i, x_j} \) to \( x_i \). This could be easily extended to higher order cases. On one hand, the \( k \)-order derivative with respect to \( \Delta_{k_1} \ldots \Delta_{k_n} \) at 0 point is

\[
g_{\Delta_{k_1} \ldots \Delta_{k_n}}(0) = \left[ \frac{\partial^k f(x + \alpha \Delta)}{\partial \Delta_{k_1} \ldots \Delta_{k_n}} \right]_{\Delta=0} = \int_0^1 \frac{\partial f(x + \alpha \Delta)}{\partial \Delta} \, d\alpha
\]

where the vector \( k = [k_1, \ldots, k_n] \) satisfies that \( \sum_m k_m = k \). The coefficient of \( \Delta_{k_1} \ldots \Delta_{k_n} \) term is \( \frac{1}{k!} \left( k \right)_{k_1 \ldots k_n} g_{\Delta_{k_1} \ldots \Delta_{k_n}}(0) \), therefore the corresponding Taylor expansion term is,

\[
\frac{1}{(k_1 + 1)!} \left( k_1 \right)_{k_1 \ldots k_n} \frac{\partial f(x)}{\partial x_i} \Delta_i \ldots \Delta_n
\]

As \( a_i = -\Delta_i g(\Delta) \), then the corresponding term in \( a_i \) is,

\[
\frac{-1}{(k_1 + 1)!} \left( k_1 \right)_{k_1 \ldots k_n} \frac{\partial f(x)}{\partial x_i} \Delta_i \ldots \Delta_n
\]

(17)
We compare the term in Eq. (17) with the coefficient of \(k+1\) order term \(\Delta_kx_1\ldots \Delta_kx_n\) of \(f(\tilde{x}) - f(x)\) expanded at \(x\), i.e., \[
\frac{1}{(k+1)!} \left( \frac{\partial}{\partial x_1} \ldots \frac{\partial}{\partial x_n} \right) f(\tilde{x}) = \frac{k!}{(k+1)!} = \frac{k+1}{k+1} \]
We can obtain that when \(k_i \neq 0\), Integrated Gradients allocates
\[
\frac{k}{(k+1)!} = \frac{k!}{(k+1)!} = \frac{k}{k+1}
\]
proportion of the \(k+1\) order term \(\Delta_kx_1\ldots \Delta_kx_n\) in Taylor expansion of \(f(x)\) to feature \(x_i\). When \(k_i = k\), the aforementioned term becomes \(k+1\) order independent term, and then corresponding proportion equals to 1. □

**Proof of Theorem 7**

**Proof.** We rewrite the \(j\)-th neuron at \(l+1\) layer as \(y\) and the \(i\)-th neuron at \(l\)-th layer as \(x_i\), then we have
\[
y = \sigma(\sum_i w_{ij} x_i + b_j).
\]
We denote the above function as \(f^l(x)\). Integrated Gradients calculates the attribution of \(f^l(x)\) by,
\[
a_{i}^{IG} = \Delta_i \int_0^1 \frac{\partial f^l(x + \alpha \Delta)}{\partial x_i} d\alpha
\]
Computing the partial derivative, we obtain that,
\[
a_{i}^{IG} = \Delta_i \int_0^1 \sum_a \sum_{x_j} \frac{\partial f^l(x)}{\partial x_i} \sigma(\alpha \sum a w_{ij} x_j + c) d\alpha
\]
where \(c = \sum_i w_{ij} x_i + b_j\). Consider \(F(\alpha) = \sigma(\alpha \sum a w_{ij} x_j + c)\) as a function of \(\alpha\), whose derivative is \(\sigma'(\alpha \sum a w_{ij} x_j + c)\) \(\sum_i w_{ij} \Delta_i\). Therefore, we have,
\[
a_{i}^{IG} = \sum_i w_{ij} \Delta_i \int_0^1 (f(1) - f(0))
\]
Because \(w_{ij} \Delta_i = z_i^j - z_i^0\), we obtain,
\[
a_{i}^{IG} = \sum_i z_i^j - z_i^0 = a_{i}^{DL}.
\]
Then the conclusion holds.

**Proof of Theorem 8**

**Proof.** According to Taylor expansion, \(f(S \cup \{i\}) - f(S)\) could be decomposed as the sum of first-order term, high-order independent term and high-order interactive terms with all possible subsets \(S'\) in \(S\):
\[
f(S \cup \{i\}) - f(S) = T_i^\alpha + T_i^\gamma + \sum_{\emptyset \subset S' \subset S \setminus \{i\}} T_{S \cup \{i\}}^\gamma.
\]
Hence, the Eq. (18) can be represented as follows:
\[
a_i = \frac{1}{n} \sum_{k=0}^{n-1} \sum_{S \subset X \setminus \{i\}} \left( \frac{n-1}{|S|} \right) \left( T_i^\alpha + T_i^\gamma \right) + \frac{1}{n} \sum_{k=0}^{n-1} \sum_{S \subset X \setminus \{i\}} \left( \frac{n-1}{|S|} \right) \sum_{\emptyset \subset S' \subset S \setminus \{i\}} T_{S \cup \{i\}}^\gamma
\]
As the first-order term and high-order independent term don’t depend on the size of \(S\), we have,
\[
\frac{1}{n} \sum_{k=0}^{n-1} \sum_{S \subset X \setminus \{i\}} \left( \frac{n-1}{|S|} \right) \left( T_i^\alpha + T_i^\gamma \right) = T_i^\alpha + T_i^\gamma
\]
Then we consider how important is each player to the cooperation of \(S \cup \{x_i\}\) in Shapley value method. Specifically, how much of \(T_{S \cup \{x_i\}}^\gamma\) is allocated to \(x_i\).

We observe that \(T_{S \cup \{x_i\}}^\gamma\) would appear once in \(f(\tilde{S} \cup \{i\}) - f(\tilde{S})\), where the set \(\tilde{S}\) satisfies \(S \subseteq \tilde{S}\). With the size of \(S\) fixed, the number of sets \(\tilde{S}\) where the term contains \(T_{S \cup \{x_i\}}^\gamma\) is
\[
\frac{n-1}{|S| - |S|}.
\]
Hence, the \(T_{S \cup \{x_i\}}^\gamma\) term contained in \(a_i\) is calculated by,
\[
\frac{1}{n} \sum_{|S| = |S|} \left( \frac{n-1}{|S| - |S|} \right) T_{S \cup \{x_i\}}^\gamma
\]
We simultaneously multiply \(|S|\) with numerator and denominator, and obtain,
\[
\frac{1}{n} \left( n - 1 \right) \left( n - 1 \right) \sum_{|S| = |S|} \left( \frac{|S|}{|S|(|S| - |S|)} \right) T_{S \cup \{x_i\}}^\gamma
\]
According to Hockey-stick identity \([L] i.e., \sum_{i=r}^{n} i = \binom{n+1}{r+1}\), we can obtain that,
\[
\frac{1}{n} \left( n - 1 \right) \left( n - 1 \right) \sum_{|S| = |S|} \left( \frac{|S|}{|S|(|S| - |S|)} \right) T_{S \cup \{x_i\}}^\gamma
\]
Then the conclusion holds. □

**Proof of Theorem 9**

**Proof.** We mainly focus on \(a_{i}^{DL+}\), the proof of \(a_{i}^{DL-}\) can be derived analogously. Assume feature \(x_i\) belongs to subset \(A^+\), then \(a_{i}^{DL+}\) for DeepLIFT RescaleCancel \([14]\) is,
\[
a_{i}^{DL+} = \frac{\Delta z_i^+ \Delta y_i^+}{\Delta z^+} a_{ij}^{(l+1)} + \frac{\Delta z_i^+ \Delta y_i^+}{\Delta z^+} a_{ij}^{(l+1)}
\]
where \(\Delta y_i^+ = \frac{1}{2}(\sigma(\tilde{z} + \Delta z^+ - \sigma(\tilde{z}))\) and \(\Delta y_i^- = \frac{1}{2}(\sigma(\tilde{z} + \Delta z^+ - \sigma(\tilde{z}^-))\) represents the first and second term of \(\Delta y^+\), respectively. We can observe that \(\Delta y_i^+\) is half of the output difference caused by input difference \(\Delta z^+\) (expand at \(\tilde{z}\)), then according to Theorem 7 for DeepLIFT Rescale,
\[
\frac{\Delta z_i^+}{\Delta z^+} \Delta y_i^+ = \frac{1}{2}(T_i^\alpha + T_i^\gamma + a_{i,A}^{(IG)})
\]
where \( a_i^{\gamma, A^+} \) represents the assignment from the interactive terms between \( x_i \) and other features in \( A^+ \). Next we expand \( \Delta y_2^+ \) at \( z \), and we have,

\[
\Delta y_2^+ = \frac{1}{2} \left( (\sigma(\tilde{z} + \Delta z) - \sigma(\tilde{z})) + (\sigma(\tilde{z}) - \sigma(\tilde{z} + \Delta z^-)) \right)
\]

\[
= \frac{1}{2} \left( \sum_i T_i^+ + \sum_i T_i^- + \sum_{B^+ \subseteq A^+} T_{B^+}^- + \sum_{B^- \subseteq A^-} T_{B^-}^+ \right).
\]

Similarly, \( \Delta y_2^- \) is also half of the output difference caused by input difference \( \Delta z^- \), then,

\[
\frac{\Delta z^+}{\Delta z^+} \Delta y_2^+ = \frac{1}{2} (T_i^+ + T_i^- + a_i^{\gamma, A^+} + a_i^{\gamma, A^-} (IG)).
\]

Here, \( a_i^{\gamma, A^-} \) represents the assignment from the interactive terms between \( x_i \) and all features in subset \( A^- \).

Therefore, for the features \( x_i \) in the positive subset \( A^+ \), the attribution is expressed as:

\[
a_i^{DL^+} = \frac{\Delta z^+}{\Delta z^+} \frac{\Delta y_2^+ - \Delta y_2^-}{\Delta y} a_j^{(t+1)}
\]

\[
= (T_i^+ + T_i^- + a_i^{\gamma, A^+} + a_i^{\gamma, A^-} (IG)) \frac{a_j^{(t+1)}}{\Delta y}.
\]

Noted that the constant \( a_j^{(t+1)} \) and \( \Delta y \) does not affect the attribution result. Then the conclusion holds.

**Proof of Theorem 10**

Proof. In Deep Taylor [15], \( \Delta y^+ = \Delta y \) and \( \Delta y^- = 0 \). We firstly conduct a second-order Taylor expansion to the function \( \Delta y \), which can be naturally expanded to higher-order Taylor terms.

\[
\Delta y = \delta(z) - \delta(\tilde{z})
\]

\[
= \delta(\tilde{z} + \Delta z^+ + \Delta z^-) - \delta(\tilde{z})
\]

\[
= T_{A^+}^\alpha + T_{A^+}^\gamma + T_{A^-}^\gamma + T_{A^+}^\gamma
\]

\[
= (\sigma(\tilde{z} + \Delta z^+ + \Delta z^-) - \sigma(\tilde{z})) + (\sigma(\tilde{z}) - \sigma(\tilde{z} + \Delta z^-))
\]

\[
\Delta z^+ (T_{A^+}^\gamma + T_{A^-}^\gamma + T_{A^+}^\gamma) + \Delta z^- (T_{A^+}^\gamma + T_{A^-}^\gamma + T_{A^+}^\gamma)
\]

Similar to the derivation of Theorem 7 (DeepLIFT Rescale),

\[
\frac{\Delta z^+}{\Delta z^+} \Delta y = T_{A^+}^\alpha + T_{A^+}^\gamma + a_{i,A^+} (IG)
\]

\[
+ \frac{\Delta z^-}{\Delta z^-} (T_{A^-}^\gamma + T_{A^-}^\gamma + T_{A^+}^\gamma + T_{A^+}^\gamma)
\]

Then the conclusion holds. We can obtain that Deep Taylor allocates the extra contribution from the negative features in \( A^- \) to the positive features in \( A^+ \). □

**Proof of Theorem 11**

Proof. In LRPAβ [6], \( \Delta y^+ = \alpha \Delta y \) and \( \Delta y^- = -\beta \Delta y \), which is similar to the definition in Deep Taylor Decomposition (\( \Delta y^+ = \Delta y, \Delta y^- = 0 \)). Only a constant factor is multiplied in LRPAβ. Therefore, the proof is almost the same as the proof of Theorem 10 □

**Proof of Theorem 12**

According to Theorem 5, we can obtain that the attribution \( a_{i,IG}^\gamma (\Delta) \) of Integrated Gradients w.r.t baseline point \( \tilde{x} \) can be reformulated as,

\[
a_{i,IG}^\gamma (\Delta) = T_i^\alpha (\Delta) + T_i^\gamma (\Delta) + a_i^{\gamma} (\Delta).
\]

The attribution \( a_{i,EG}^\gamma \) of feature \( x_i \) in Expected Gradients is an integral of attribution \( a_{i,IG}^\gamma (\Delta) \) over baseline distribution \( p_D (\tilde{x}) \). We represent the distribution of \( \Delta, p_D (\Delta) (as \Delta = \tilde{x} - x) \). Then the attribution of Expected Gradients \( a_{i,EG}^\gamma \) could be rewritten as

\[
a_{i,EG}^\gamma = \int \frac{a_{i,IG}^\gamma (\Delta) p_D (\tilde{x}) d\tilde{x}}{\Delta}
\]

\[
= \int \frac{a_{i,IG}^\gamma (\Delta) p_D (\Delta) d\Delta}{\Delta}
\]

\[
= \int \frac{T_i^\alpha (\Delta) + T_i^\gamma (\Delta) + a_i^{\gamma} (\Delta)}{\Delta} d\Delta.
\]

Hence, Expected Gradients averages the first-order term, high-order independent and high-order interactive terms of multiple \( \Delta \). This could avoid the case in which Integrated Gradients is dominated by a specific \( \Delta \).

**APPENDIX B**

**MORE EXPERIMENTAL RESULTS**

Appendix two text goes here.

**ACKNOWLEDGMENT**

The authors would like to thank...

**ACKNOWLEDGMENTS**

The authors would like to thank...

Michael Shell Biography text here.

John Doe Biography text here.

Jane Doe Biography text here.