IDGI: A Framework to Eliminate Explanation Noise from Integrated Gradients

Ruo Yang, Binghui Wang, and Mustafa Bilgic
Department of Computer Science, Illinois Institute of Technology, Chicago, IL, USA
ryang23@hawk.iit.edu, bwang70@iit.edu, mbilgic@iit.edu

Abstract

Integrated Gradients (IG) as well as its variants are well-known techniques for interpreting the decisions of deep neural networks. While IG-based approaches attain state-of-the-art performance, they often integrate noise into their explanation saliency maps, which reduce their interpretability. To minimize the noise, we examine the source of the noise analytically and propose a new approach to reduce the explanation noise based on our analytical findings. We propose the Important Direction Gradient Integration (IDGI) framework, which can be easily incorporated into any IG-based method that uses the Reimann Integration for integrated gradient computation. Extensive experiments with three IG-based methods show that IDGI improves them drastically on numerous interpretability metrics. The source code for IDGI is available at https://github.com/yangruo1226/IDGI.

1. Introduction

With the deployment of deep neural network (DNN) models for safety-critical applications such as autonomous driving [5–7] and medical diagnostics [10, 24], explaining the decisions of DNNs has become a critical concern. For humans to trust the decision of DNNs, not only the model must perform well on the specified task, it also must generate explanations that are easy to interpret. A series of explanation methods (e.g., gradient-based saliency/attribution map approaches [21, 22, 29, 33, 36, 38, 43, 46] as well as many that are not based on gradients [4, 11, 13, 19, 25, 27, 30, 32, 35, 39, 40, 42, 47, 48]) have been developed to connect a DNN’s prediction to its input. Among them, Integrated Gradients (IG) [43], a well-known gradient-based explanation method, and its variants [22, 46] have attracted significant interest due to their state-of-the-art explanation performance and desirable theoretical properties. However, we observe that explanation noise exists in the attribution generated by these IG methods (please see Fig. 1). In this research, we investigate IG-based methods, study the explanation noise introduced by these methods, propose a frame-

Figure 1. Saliency/attribution map of the existing IG-based methods and those with our method on explaining the prediction from InceptionV3 model. Our method significantly reduces the noise in the saliency map created by these IG-based methods.

1 [22] mentions the accuracy of integration is also a reason to generate the noise, but this is not the focus of existing IG methods and this paper.
nates the noise by examining the integration directions from the explanation perspective.

Specifically, we investigate each computation step in the Riemann Integration and then theorize about the noises’ origin. Each Riemann integration calculation integrates the gradient in the original direction—it first computes the gradient with respect to the starting point of the current path segment and then multiplies the gradient by the path segment. We show that the original direction can be divided into an important direction and a noise direction. We theoretically demonstrate that the true gradient is orthogonal to the noise direction, resulting in the gradient’s multiplication along the noise direction having no effect on the attribution. Based on this observation, we design a framework, termed Important Direction Gradient Integration (IDGI), that can eliminate the explanation noise in each step of the computation in any existing IG method. Extensive investigations reveal that IDGI reduces noise significantly when evaluated using state-of-the-art IG-based methods.

In summary, our main contributions are as follows:

• We propose the Important Direction Gradient Integration (IDGI), a general framework to eliminate the explanation noise in IG-based methods, and investigate its theoretical properties.

• We propose a novel measurement for assessing the attribution techniques’ quality, i.e., AIC and SIC using MS-SSIM. We show that this metric offers a more precise measurement than the original AIC and SIC.

• Our extensive evaluations on 11 image classifiers with 3 existing and 1 proposed attribution assessment techniques indicate that IDGI significantly improves the attribution quality over the existing IG-based methods.

2. Background

2.1. Integrated Gradient and its Variants

Integrated Gradients (IG) [43]. Given a classifier $f$, class $c$, and input $x$, the output $f_c(x)$ represents the confidence score (e.g., probability) for predicting $x$ to class $c$. IG computes the importance/attribution per feature (e.g., a pixel in an image) by calculating the line integral between the reference point $x'$ and image $x$ in the vector field that the model creates, where the vector field is formed by the gradient of $f_c(x)$ with respect to the input space. Formally, for each feature $i$, the IG is defined as below:

$$I^{IG}_i(x) = \int_{0}^{1} \frac{\partial f_c(\gamma^IG(\alpha))}{\partial \gamma^IG(\alpha)} \frac{\partial \gamma^IG(\alpha)}{\partial \alpha} d\alpha,$$

where $\gamma^IG(\alpha), \alpha \in [0, 1]$ is the parametric function representing the path from $x'$ to $x$, e.g., $\gamma^IG(0) = x'$, $\gamma^IG(1) = x$. Specifically, $\gamma^IG$ is a straight line that connects $x'$ and $x$.

Guided Integrated Gradients (GIG) [22]. Kapishnikov et al. [22] claim that DNN’s output shape has a high degree of curvature; hence, the larger-magnitude gradients from each feasible point on the path would have a significantly larger effect on the final attribution values. To address it, they propose GIG, an adaptive path method for noise reduction. Specifically, instead of integrating gradients following the straight path as IG does, GIG initially seeks a new path to avoid high-gradient directions as below:

$$\gamma^{GIG} = \arg \min_{\gamma \in \Gamma} \sum_{i=1}^{N} \int_{0}^{1} \left| \frac{\partial f_c(\gamma(\alpha))}{\partial \gamma(\alpha)} \frac{\partial \gamma(\alpha)}{\partial \alpha} \right| d\alpha,$$

where $\Gamma$ is the set containing all possible path between $x'$ and $x$. After finding the optimal path $\gamma^{GIG}$, GIG uses it and computes the attribution values similar to IG. Formally,

$$I^{GIG}_i(x) = \int_{0}^{1} \frac{\partial f_c(\gamma^{GIG}(\alpha))}{\partial \gamma^{GIG}(\alpha)} \frac{\partial \gamma^{GIG}(\alpha)}{\partial \alpha} d\alpha.$$  (3)

Blur Integrated Gradients (BlurIG) [46]. Xu et al. [46] propose BlurIG, which integrates the IG into frequency domain as opposed to the original image domain. In other words, BlurIG takes into account the path produced by sequentially blurring the input with a Gaussian blur filter. Specifically, suppose the image has the 2D shape $M \times N$, and let $x(p, q)$ represent the value of the image $x$ at the location of pixels $p$ and $q$. The discrete convolution of the input signal with a 2D Gaussian kernel with variance $\alpha$ is thus defined as follows:

$$\gamma^{BlurIG} := L(x, p, q, \alpha) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \frac{1}{\pi \alpha} e^{-\frac{x^2 + y^2}{\alpha^2}} x(p - m, q - n)$$

Then, the final BlurIG computation is as below:

$$I^{BlurIG}_{p,q}(x) = \int_{0}^{\infty} \frac{\partial f_c(\gamma^{BlurIG}(\alpha))}{\partial \gamma^{BlurIG}(\alpha)} \frac{\partial \gamma^{BlurIG}(\alpha)}{\partial \alpha} d\alpha.$$  (5)

2.2. Riemann Integration

Existing IG-based methods (IG: Eq. (1), GIG: Eq. (3), and BlurIG: Eq. (5)) all approximate the line integral numerically using the Riemann Integration. Specifically, as shown in Eq. (6), they discretize the path between $x'$ and $x$ into $N$, a large and finite number, small piece-wise linear segments, and aggregate the value of multiplication between the gradient vectors and the small segments:

$$f_c(x) - f_c(x') = \lim_{N \to \infty} \sum_{j=0}^{N} \frac{f_c(x_j)}{\partial x_j} (x_{j+1} - x_j)$$

$$= \sum_{i=0}^{n} I_i(x) = \int_{0}^{1} \frac{\partial f_c(\gamma(\alpha))}{\partial \gamma(\alpha)} \frac{\partial \gamma(\alpha)}{\partial \alpha} d\alpha.$$  (6)
Figure 2. Illustration of IDGI. The red, green, and purple lines illustrate the path corresponding to IG, GIG, and BlurIG, respectively, where both IG and GIG need a reference point as manual input but BlurIG does not. Riemann Integration computes the integral by multiplying the blue and black vectors at each calculation step. However, each black vector can be linearly decomposed of blue and brown vectors. Since the direction of the blue vector, i.e. gradient, represents the maximum rate of change for the function value, we consider it to be the most important direction. Instead of multiplying the blue vector by the black vector, we propose computing the integral using the blue vector alone. In addition to the fact that the multiplication of the blue and brown vectors has no effect on the function’s value, we assert that this integration also creates the noise.

In other words, regardless of whichever approach (IG, GIG, or BlurIG) is used for calculating the attributions, the final attribution map is produced from Riemann Integration in all IG-based algorithms. As the algorithm approximates the integration discretely, the approximation itself contains numerical inaccuracies relative to the theoretical real values. However, we do not concentrate on eliminating numerical errors; rather, we discovered that each Riemann Integration step creates noise from an explanation perspective (please see Fig. 2). Specifically, each path segment has a noise direction where gradient integration with that direction contributes nothing to output values, which indicates the attribution values generated with this direction are noisy.

3. Our Framework: Important Direction Gradient Integration (IDGI)

In this section, we first describe the concept of important direction and the noise that arises from the gradient with the noise direction to the attributions. Then, we formally introduce the Important Direction Gradient Integration (IDGI), a framework that only leverages the gradient with the important direction. We highlight that IDGI can be applied to any IG-based method. Finally, we discuss the theoretical properties of IDGI.

3.1. Important Direction and Noise Direction

Important Direction. Given the point \( x_j = \gamma(\alpha_j) \) and its next point \( x_{j+1} = \gamma(\alpha_{j+1}) \) on the path from reference point \( x' \) to the input point \( x \), IG-based methods compute the gradient, \( g \), of \( f_c(x_j) \) with respect to \( x \) and utilize Riemann integration to multiply element-wisely with the direction (original direction) from \( x_j \) to \( x_{j+1} \) (please see Fig. 2). Based on the Riemann integration principle, when \( N \) increases, the sum of the multiplication result accurately estimates the differences in the function values, \( f_c(x_{j+1}) - f_c(x_j) \). In terms of the explanation, the multiplication result for this step indicates the contribution to the change in the value of the function from \( f_c(x_j) \) to \( f_c(x_{j+1}) \). In other words, attribution values at this step explain why the prediction score moves from \( f_c(x_j) \) to \( f_c(x_{j+1}) \).

The unit direction vector of the gradient at step \( j \), i.e., \( \frac{\partial f_c}{\partial x} \), indicates the direction of the fastest increase of the function \( f_c \) at \( x_j \). That is, moving the point \( x_j \) along the direction \( g \) changes \( f_c \) the most. We refer to the direction \( \frac{\partial f_c}{\partial x} \) as the important direction. In general, the gradient of the function value \( f_c \) at each point in space defines the conservative vector field, where an infinite number of hyperplanes exist, and each hyperplane contains all points with the same function output value. For instance, \( x_j \) resides on the hyperplane \( h_j \) if all the points \( x_{h_j} \in h_j \) have the same function value \( f_c(x_j) \), i.e., \( f_c(x_{h_j}) = f_c(x_j) \), \( \forall x_{h_j} \in h_j \). In the conservative vector field, separate hyperplanes never intersect, which means that each point has its own projection point with regard to the other hyperplanes. Moreover, to identify the projections, one may move the point from its original hyperplane toward the next hyperplane in which the moving direction is the same as the gradient’s direction \( \frac{\partial f_c}{\partial x} \).

Similarly, \( x_{j+1} \) stays on the hyperplane \( h_{j+1} \) where \( f_c(x_{h_{j+1}}) = f_c(x_{j+1}) \), \( \forall x_{h_{j+1}} \in h_{j+1} \). For point \( x_j \), if one moves \( x_j \) along the important direction, there exists an unique projection point \( x_{j_p} \) on the hyperplane \( h_{j+1} \) where \( f_c(x_{j_p}) = f_c(x_{j+1}) \). As we illustrate in Theorem 1, while the attribution for each feature \( i \) computed from the original direction \( (x_{j+1} - x_j) \) and important direction \( (x_{j_p} - x_j) \) could be different, the change in the value of the function \( f_c \), which are the prediction values to be explained are the same since \( x_{j+1} \) and \( x_{j_p} \) are on the same hyperplane \( h_{j+1} \).

Theorem 1 Given a function \( f_c(x) : R^n \rightarrow R \), points \( x_j, x_{j+1}, x_{j_p} \in R^n \), then the gradient of the function with respect to each point in the space \( R^n \) forms the conservative vector fields \( \nabla f_c \) and further define the hyperplane \( h_j = \{ x : f_c(x) = f_c(x_j) \} \) in \( R^n \). Assume the Riemann Integration accurately estimates the line integral of the vector field \( \nabla f_c \) from points \( x_j \) to \( x_{j+1} \) and \( x_{j_p} \), e.g. \( \int_{x_j}^{x_{j+1}} \frac{\partial f_c(x)}{\partial x} dx \approx \frac{\partial f_c(x)}{\partial x} | x_j \rightarrow x_{j+1} \), \( x_j \in h_j, x_{j_p} \in h_{j+1} \). Then:

\[
\int_{x_j}^{x_{j+1}} \frac{\partial f_c(x)}{\partial x} dx \approx \int_{x_j}^{x_{j_p}} \frac{\partial f_c(x)}{\partial x} dx.
\]

Noise Direction. For step \( j \), any original direction can be decomposed into a combination of important direction and
The prediction value changes from \( O \), the integration of the first path, i.e., from \( x \) stays the same regardless of which path is chosen. This suggests that the second path attributions (from \( x \) to \( x_{j+1} \)) do not account for any changes in prediction score. We refer to this direction of the second path as the noise direction and argue that the integration with this direction adds noise to the attribution since it explains zero contribution for changes in the prediction score.

Since \( f_c(x_{j+1}) \approx f_c(x_j) \) (as we demonstrated in Theorem 1), the integration of the first path, i.e., from \( x_j \) to \( x_{j+1} \), of the two-parts-path offers the full attributions that explain the prediction value changes from \( f_c(x_j) \) to \( f_c(x_{j+1}) \). This suggests that the second path attributions (from \( x_j \) to \( x_{j+1} \)) do not account for any changes in prediction score values. We refer to this direction of the second path as the noise direction and argue that the integration with this direction adds noise to the attribution since it explains zero contribution for changes in the prediction score.

### 3.2. IDGI Algorithm

In order to reduce the noise from the attributions, we propose the Important Direction Gradient Integration (IDGI) framework. Suppose we are given an input \( x \), target class \( c \), classifier \( f \), and a given path \( \text{path} : [x', \ldots, x_j, \ldots, x] \) from any IG-based method. Similar to IG-based approaches, IDGI first calculates the gradient \( g \) with respect to the current point at each Riemann integration step. IDGI then determines the unit important direction vector of \( g \) as \( \frac{g}{|g|} \) and the step size \( \frac{f_c(x_{j+1})-f_c(x_j)}{|g|} \) rather than multiplying \( g \) with the distance \( x_{j+1} - x_j \). The step size determines how much of the unit direction vector should be applied and ensures that the current step explains the change in probability from \( f_c(x_{j+1}) \) to \( f_c(x_j) \). In other words, the projection of \( x_j \) onto the hyperplane \( h_{j+1} \), formed as \( x_{j_p} = x_j + \frac{g_j}{|g|} \cdot \frac{f_c(x_{j+1})-f_c(x_j)}{|g|} \), has the same function value as point \( x_{j+1} \), i.e., \( f_c(x_{j+1}) = f_c(x_{j_p}) \), since both \( x_{j+1} \) and \( x_{j_p} \) reside on the hyperplane \( h_{j+1} \). Finally, IDGI aggregates the computations from each step and forms the final attributions to interpret the prediction score of \( f_c(x) \). The pseudo-code of IDGI is described in Algorithm 1.

#### Algorithm 1 Important Direction Gradient Integration

Inputs: \( x, f, c, \text{path} : [x', \ldots, x_j, \ldots, x] \)

1: \( I_{IDGI}^{1} = 0 \)
2: for \( x_j \in \text{path} \) do
3: \( d = f_c(x_{j+1}) - f_c(x_j) \)
4: \( g = \frac{\partial f_c(x_j)}{\partial x_j} \)
5: \( I_{IDGI}^{1} += \frac{g_j}{|g|} \cdot \frac{f_c(x_{j+1})-f_c(x_j)}{|g|} \)
6: end for
7: return \( I_{IDGI}^{1} \)

### 3.3. Theoretical Properties of IDGI

Sundararajan et al. [43] introduce several axioms that any explanation method is expected to adhere to. These axioms include Completeness, Sensitivity(a, b), Implementation Invariance, Linearity, and Symmetry preserving. As we discuss below, IDGI satisfies all of them except Linearity.

**Completeness.** IDGI clearly satisfies this axiom since the computation for \( j \)th step in IDGI computes the attribution for \( f_c(x_{j+1}) - f_c(x_j) \) exactly. This also indicates that IDGI satisfies Sensitivity-N [3].

**Sensitivity(a).** IDGI satisfies Sensitivity(a) at each computation step \( j \) (from \( x_j \) to \( x_{j+1} \)) if the underlying IG-based method to which IDGI is applied satisfies the axiom. Consider a case that during the \( j \)th computation step, the baseline at this step refers to \( x_j \), input refers to \( x_j \) and the two points differ only at \( i \)th feature, e.g., \( x'_j \neq x_{j+1} \), \( \forall q = i \) and \( x'_j \neq x_{j+1} \), \( \forall q \neq i \). Also, assume the predictions vary due to the \( i \)th feature differences, e.g., \( f_c(x_j) \neq f_c(x_{j+1}) \). If the underlying IG-based method satisfies Sensitivity(a), then the \( i \)th of the computed gradient \( g \) at this step is non-zero, i.e., \( g_i \neq 0 \). Then IDGI assigns non-zero value to the attribution of feature \( i \) at step \( j \) since \( g_i \neq 0 \times (f_c(x_{j+1}) - f_c(x_j)) \neq 0 \). From \( x' \) to \( x \), IDGI satisfies Sensitivity(a) if the underlying IG-based method, to which IDGI is applied, satisfies the axiom and the function evaluation on the giving path \( p \) is strictly monotonic, e.g., \( f_c(x_{j+1}) > f_c(x_j) \) or \( f_c(x_{j+1}) < f_c(x_j), \forall x_j \in p \). Since the underlying IG-based method satisfies Sensitivity(a), it indicates that there exists at least one gradient \( g \) which has a non-zero value for \( i \)th feature, i.e., \( g_i \neq 0 \), during all the computation steps. Then, the strictly monotonic property guarantees the non-zero value is captured in the final attribution for \( i \)th feature. Also, since the square value of \( g_i \) is computed during each step, the strictly monotonic property assures that any attribution value added to the final attribution contributes in the same direction rather than canceling each other. Furthermore, intuitively, the strictly monotonic property of the function evaluation on the giving path \( p \) is expected since the changing of one feature in one direction is expected to impact the output of the function in one direction too, e.g. increasing the prediction value.

**Sensitivity(b).** IDGI satisfies this axiom as long as the underlying IG-based method satisfies it. If the IG-based method satisfies Sensitivity(b), then any variable/feature \( i \) that the function \( f_c \) does not rely on will have zero gradient value everywhere, i.e., \( g_i = 0 \). Clearly, the final attribution of IDGI also assigns the value of zero to such a feature \( i \) since \( g_i \) is zero in each step of the computation.

**Implementation Invariance.** Kapishnikov et al. [22] showed an attribution method that depends only on function gradients and is independent of the network’s underlying structure satisfies the Implementation Invariance axiom.
Thus, IDGI preserves the invariance as long as the underlying IG-based method is independent of the network’s topology and solely depends on the gradients of the functions.

**Linearity.** Given a linearly combined network $f_3 = a \times f_1 + b \times f_2$, Linearity requires that the explanation method assign attribution values as a weighted combination of attribution from $f_1$ and $f_2$, i.e., $\text{IG}_{f_3}(x) = a \times \text{IG}_{f_1}(x) + b \times \text{IG}_{f_2}(x)$. IDGI does not satisfy this requirement. However, in practice, we often try to explain the predictions of a complex nonlinear function, such as DNNs, instead of the linear composition of models.

**Symmetry preserving.** An attribution method is Symmetry preserving if, 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 [43]. IDGI satisfies this axiom only if the underlying IG-based method satisfies it.

### 4. Experimental Methodology and Results

#### 4.1. Experimental Setup

**Baselines.** We use four gradient-based methods as baselines. Since the Integrated Gradients (IG) [43], Guided Integrated Gradients (GIG) [22], and Blur Integrated Gradients (BlurIG) [46] offer distinct paths from the reference point to the image of interest, our approach could be applied independently to any of them because it is orthogonal to any given path. Additionally, we also include the Vanilla Gradient (VG) [36], which takes the gradient of the output $f_c(c)$ with respect to $x$. We use the original implementations\(^2\) with default parameters in the authors’ code for IG, GIG, and BlurIG, and implement our method as an additional module that interfaces with the original implementations. Then, same as [22], we use a step size of 200 as the parameter needed to compute the Riemann integral for all methods except VG since it doesn’t use Riemann Integration. As is the common practice, we use the black image as the reference point for IG and GIG.

**Models.** We use the TensorFlow (2.3.0) [1] pre-trained models: DenseNet\{121, 169, 201\} [18], InceptionV3 [44], MobileNetV2 [17], ResNet\{50, 101, 151\}V2 [16], VGG\{16, 19\} [37], and Xception [8].

**Dataset.** Following the research [21, 22, 29, 46], we use the Imagenet [12] validation dataset, which contains 50K test samples with labels and annotations. We test the explanation methods for each model on images that the model predicted the label correctly. Hence, the number of test images varies from 33K to 39K corresponding to different models.

**Evaluation metrics.** We use numerous metrics to quantitatively evaluate our method and compare it with baselines: Insertion score [29, 30], the Softmax information curves (SIC), the Accuracy information curves (AIC) [21, 22], and three Weakly Supervised Localization [9, 21, 46] metrics. We also introduce a modified version of SIC and AIC with MS-SSIM information level. We implement all the evaluation metrics as they were introduced in the previous works, and we discuss the implementation details in Sec. 4.3-4.6.

#### 4.2. Qualitative Check

Fig. 3 shows a sample of observations for all baselines and IDGI with all models. Compared to baseline methods, the outcome of our method demonstrates the relevant pixels are more concentrated in the bison region of the original image. Also, the noises are less in other regions. However, qualitative and visual inspections are often subjective and hence we focus more on the quantitative metrics in the rest of the experiments.

#### 4.3. Insertion Score

In this investigation, we assess our explanation approaches with the Insertion Score from prior works [29, 30]. For each test image, starting with a blank image, the in-

\(^2\)https://github.com/PAIR-code/saliency

---

**Figure 3.** Saliency map comparisons between different methods and different models (each row). Each model correctly predicts the image as bison. Left-to-right: Original image, VG, IG, IG + IDGI, GIG, GIG + IDGI, BlurIG, and BlurIG + IDGI. Methods with IDGI focus more on the region of the bison.
Models

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Models</th>
<th>IG-based Methods</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertion Score with Probability (%)</td>
<td>DenseNet201</td>
<td>293.241</td>
<td>904.354</td>
</tr>
<tr>
<td></td>
<td>DenseNet169</td>
<td>350.503</td>
<td>362.421</td>
</tr>
<tr>
<td></td>
<td>DenseNet201</td>
<td>329.457</td>
<td>341.403</td>
</tr>
<tr>
<td></td>
<td>InceptionV3</td>
<td>348.478</td>
<td>368.406</td>
</tr>
<tr>
<td></td>
<td>MobileNetV2</td>
<td>203.346</td>
<td>231.291</td>
</tr>
<tr>
<td></td>
<td>ResNet50V2</td>
<td>338.431</td>
<td>364.403</td>
</tr>
<tr>
<td></td>
<td>ResNet151V2</td>
<td>388.480</td>
<td>388.418</td>
</tr>
<tr>
<td></td>
<td>VGG16</td>
<td>217.338</td>
<td>212.265</td>
</tr>
<tr>
<td></td>
<td>VGG19</td>
<td>232.342</td>
<td>224.273</td>
</tr>
<tr>
<td></td>
<td>Xception</td>
<td>334.487</td>
<td>367.475</td>
</tr>
</tbody>
</table>

Table 1. Insertion score for different models with explanation methods. We report the area under the curves formed by the originally predicted probability of modified images and the normalized probability (probability ratio: the predicted probability of modified images over the predicted probability of original images.) IDGI improves all methods for all models.

4.4. SIC and AIC

We next evaluate the explanation methods using the Softmax information curves (SIC) and the Accuracy information curves (AIC) [21]. The evaluation method starts with a blurred version of the given test image and then puts back the most important pixel’s values as determined by the explanation approach, resulting in a bokeh image. Moreover, an information level is computed for each bokeh image based on explanation technique, and we create the model performance curves as the mean model performance on all the reconstructed images at each level. We report the insertion scores in Tab. 1. As expected, the VG always has the lowest score across the experiments with different models. IDGI improves the Insertion Score drastically, in all cases, for all the IG-based methods.

4.5. SIC and AIC with MS-SSIM

In the third experiment, instead of the Normalized entropy as the information level, we use the Multi-scale Structural Similarity (MS-SSIM) [45]. MS-SSIM is a well-studied [20, 26, 28] image quality evaluation method that analyzes the structural similarity of two images. It is a multi-scale variation of a well-defined perceptual similarity measure that seeks to dismiss irrelevant features (based on human perspective) of a picture [28]. MS-SSIM values range from 0.0 to 1.0; images with greater MS-SSIM values are perceptually more similar. Fig. 4 shows the distribution of

---

Figure 4. Modified distribution of bokeh images over MS-SSIM and Normalized Entropy [21]. We produce 25 bokeh pictures for each test instance based on the various attribution maps and calculate the MS-SSIM and Normalized Entropy as the information level for each bokeh image. We show the modified bokeh images of the test instance for Resnet151V2. Information level with MS-SSIM has a more evenly distribution of bokeh images than Normalized Entropy. Other models have similar trends.

Figure 5. AIC and SIC (median of normalized softmax), from InceptionV3 model, over the distribution of Normalized Entropy (Top row) and MS-SSIM (Bottom row). The area under the curve indicates IDGI provides significant improvement from all the IG-based methods. As expected, the saliency maps from the Vanilla gradient have the lowest score in all the experiments.

All modified images over MS-SSIM and the Normalized Entropy, where MS-SSIM distributed the bokeh images more evenly over the bins. That is, each bin is expected to have more samples to represent the true model performance conditioned on that similarity. The effect can be also observed in Fig. 5 where the performance for the small Normalized Entropy group has high variance due to not having enough images for that group bin. In contrast, the measurement with MS-SSIM is a much smoother curve. Hence, we propose to use MS-SSIM for estimating the information of each bokeh image rather than Normalized Entropy.

The experimental settings are the same as in the previous section except we replaced the Normalized Entropy with the MS-SSIM score as the information level. As demonstrated in Tab. 3, our approach significantly improves modified AIC and SIC for all IG-based methods.

4.6. Weakly Supervised Localization

We utilize the quantitative assessments in [9, 21, 46]. The evaluation computes ROC-AUC, F1, and MAE (mean absolute error) of the created saliency mask by considering pixels inside the annotation to be positive and pixels outside to be negative, given an annotation area. Specifically, we calculate the optimal F1 and MAE of each saliency map for each image by finding the best threshold for attribution values provided by different explanation methods. Tab. 4

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Models</th>
<th>IG-based Methods</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IG+Ours</td>
<td></td>
</tr>
<tr>
<td>AUC</td>
<td>DenseNet1121</td>
<td>229</td>
<td>.305</td>
</tr>
<tr>
<td></td>
<td>DenseNet169</td>
<td>241</td>
<td>.314</td>
</tr>
<tr>
<td></td>
<td>DenseNet201</td>
<td>254</td>
<td>.323</td>
</tr>
<tr>
<td></td>
<td>InceptionV3</td>
<td>264</td>
<td>.333</td>
</tr>
<tr>
<td></td>
<td>MobileNetV2</td>
<td>179</td>
<td>.259</td>
</tr>
<tr>
<td></td>
<td>ResNet50V2</td>
<td>225</td>
<td>.277</td>
</tr>
<tr>
<td></td>
<td>VGG16</td>
<td>205</td>
<td>.271</td>
</tr>
<tr>
<td></td>
<td>VGG19</td>
<td>211</td>
<td>.275</td>
</tr>
<tr>
<td></td>
<td>Xception</td>
<td>281</td>
<td>.362</td>
</tr>
<tr>
<td>SIC</td>
<td>DenseNet1121</td>
<td>184</td>
<td>.263</td>
</tr>
<tr>
<td></td>
<td>DenseNet169</td>
<td>205</td>
<td>.286</td>
</tr>
<tr>
<td></td>
<td>DenseNet201</td>
<td>212</td>
<td>.286</td>
</tr>
<tr>
<td></td>
<td>InceptionV3</td>
<td>211</td>
<td>.287</td>
</tr>
<tr>
<td></td>
<td>MobileNetV2</td>
<td>126</td>
<td>.204</td>
</tr>
<tr>
<td></td>
<td>ResNet50V2</td>
<td>196</td>
<td>.254</td>
</tr>
<tr>
<td></td>
<td>ResNet101V2</td>
<td>210</td>
<td>.265</td>
</tr>
<tr>
<td></td>
<td>ResNet151V2</td>
<td>221</td>
<td>.282</td>
</tr>
<tr>
<td></td>
<td>VGG16</td>
<td>163</td>
<td>.234</td>
</tr>
<tr>
<td></td>
<td>VGG19</td>
<td>173</td>
<td>.240</td>
</tr>
<tr>
<td></td>
<td>Xception</td>
<td>223</td>
<td>.312</td>
</tr>
</tbody>
</table>

Table 3. Area under the curve for SIC and AIC for 11 models. The information level is MS-SSIM. IDGI shows improvement for all three IG-based methods across all experiments.
Table 4. F1, ROC-AUC, and MAE scores. IG-based methods are improved by IDGI across all metrics and eleven models.

5. Related Work

Research on explanations for machine learning models has garnered a significant amount of attention since the development of expert models of the 1970s [2, 15]. Recently, with the increased use of Deep Neural Networks, several papers have focused on explaining the decisions of these black box models. One possible approach is based on Shapley values [14, 34]. The Shapley value was proposed to represent the contribution value of each player in the cooperative games to the outcome of the game. From the explanation perspective, the Shapley value based methods [40] computed for each feature how much it contributes to the prediction score when it’s considered alone compared to the rest of the features. The Shapley value of a feature is its true attribution value. However, calculating Shapley values is intractable when the input dimension is large. Several methods approximate the Shapley values. These include KernelSHAP [25], BShap [42], and FastShap [19].

Another strategy for explaining models’ decisions is input perturbation. Input perturbation methods work by manipulating the input and observing its effect on the output; this process is often repeated many times to produce the general behavior of the model’s prediction on that input. For example, LIME [32] approximates the decision for an input by fitting a linear model around the neighborhood of the input, where the neighborhood is generated through perturbations. The similar idea of manipulating the input is utilized by RISE [30] and several other papers [11, 13, 48].

For deep neural networks, in addition to the above strategies, several have also focused on methods of backpropagation to assign attribution to the input values. For example, modified backpropagation based methods propagate the signal of the final prediction back to the input and assign a score for each feature in the input. Methods include Deconvnet [47], guided backpropagation [39], DeepLIFT [35] and LRP [4, 27]. These approaches propagate the modified gradient/signal, instead of the true gradient, to the input.

Using the true gradient of the prediction with respect to the input as the explanation was introduced by Simonyan et al. [36]. Similarly, Shrikumar et al. [35] propose using the element-wise product (gradient ⊗ input) with input itself instead of the gradient directly. Grad-CAM [33] utilizes gradients to produce the localization map of the important regions in the input image. A popular method, Integrated Gradients (IG), was proposed by Sundararajan et al. [43]; it computes the attribution for each feature to explain the decision of a differentiable model, e.g. DNNs. Its variants include Guided Integrated Gradients (GIG) [22] and Blur Integrated Gradients (BlurIG) [46]. XRAI [21] utilizes IG to provide interpretation at the region level instead of the pixel level. SmoothGrad [38] and AGI [29] compute repeated attribution maps for a given input; however, those approaches require more computation as the single path methods (such as IG, GIG, and BlurIG) has to be repeated multiple times for a given input. Finally, I-GOS [31] and I-GOS++ [23] find a mask that would decrease the prediction score the most when it is applied to an image; they use the output of IG in their search for the mask. Our work builds on, and is orthogonal to, the single-path IG-based methods where we reduce the noise created during the integration computation. In other words, our work can be applied to any given single-path IG-based method. Furthermore, since it works with single-path methods, it is also easy to adapt to multiple-path methods as well.

6. Conclusion

We investigate the noise source generated by Integrated Gradient (IG) and its variants. Specifically, we propose the Important Direction Gradient Integration (IDGI) framework, which can be incorporated into all IG-based explanation methods and reduce the noise in their outputs. Extensive experiments show that IDGI can drastically improve the quality of saliency maps generated by the underlying IG-based approaches.

Acknowledgements. This work of Wang was supported by Wang’s startup funding, the Cisco Research Award, and the National Science Foundation under grant No. 2216926.
References


[7] Ch enquiries of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 1


[31] Zhongang Qi, Saeed Khorram, and Fuxin Li. Visualizing deep networks by optimizing with integrated gradients. In *CVPR Workshops*, volume 2, pages 1–4, 2019. 8


[34] L Shapley. Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, page 343, 1953. 8


