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 framework to remove the explanation noise, and empirically validate the effectiveness of our approach.

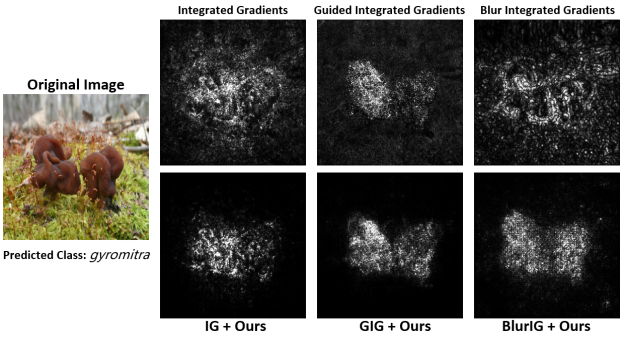

Figure 1. Saliency/attribution map of the existing IG-based methods and those with our method on explaining the prediction from *InceptionV3* model. Our method significantly reduces the noise in the saliency map created by these IG-based methods.

A few recent IG-based methods (e.g., [38] [22], [46], [41]) have been proposed to address the noise issue. Kapishnikov et al. [22] provide the following main reasons[1] that could generate the noise: 1) DNN model's shape often has a high curvature; and 2) The choice of the reference point impacts explanation. They propose Guided Integrated Gradients (GIG) [22], which tackles point #1 by iteratively finding the integration path that tries to avoid the high curvature points in the space. Blur Integral Gradients [46], on the other hand, shows that the noise could be reduced by finding the integration path through the frequency domain instead of the original image domain. Formally, it finds the path by successively blurring the input via a Gaussian blur filter. Sturmfels et al. [41] tackle point #2 by performing the integration from multiple reference points, while Smilkov et al. [38] aggregate the attribution with respect to multiple Gaussian noisy inputs to reduce the noise. Nevertheless, all IG-based methods share a common point in that they compute the integration of gradients via the Riemann integral. *We highlight that, the integration calculation by the existing methods fundamentally introduces the explanation noise.* To this end, we offer a general solution that elimi-

[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:

$$I_i^{IG}(x) = \int_0^1 \frac{\partial f_c(\gamma^{IG}(\alpha))}{\partial \gamma_i^{IG}(\alpha)} \frac{\partial \gamma_i^{IG}(\alpha)}{\partial \alpha} d\alpha, \tag{1}$$

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, γ^{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:

$$\gamma^{GIG} = \arg\min_{\gamma \in \Gamma} \sum_{i=1}^{N} \int_0^1 |\frac{\partial f_c(\gamma(\alpha))}{\partial \gamma_i(\alpha)} \frac{\partial \gamma_i(\alpha)}{\partial \alpha}| d\alpha, \tag{2}$$

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

$$I_i^{GIG}(x) = \int_0^1 \frac{\partial f_c(\gamma^{GIG}(\alpha))}{\partial \gamma_i^{GIG}(\alpha)} \frac{\partial \gamma_i^{GIG}(\alpha)}{\partial \alpha} d\alpha. \tag{3}$$

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 α is thus defined as follows:

$$\begin{aligned} \gamma^{BlurIG} &::= L(x,p,q,\alpha) \\ &= \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \frac{1}{\pi\alpha} e^{-\frac{q^2+p^2}{\alpha}} x(p-m, q-n) \end{aligned} \tag{4}$$

Then, the final BlurIG computation is as below:

$$I_{p,q}^{BlurIG}(x) = \int_{\infty}^{0} \frac{\partial f_c(\gamma^{BlurIG}(\alpha))}{\partial \gamma_{p,q}^{BlurIG}(\alpha)} \frac{\partial \gamma_{p,q}^{BlurIG}(\alpha)}{\partial \alpha} d\alpha, \tag{5}$$

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:

$$\begin{aligned} f_c(x) - f_c(x') &= \lim_{N \to \infty} \sum_{j=0}^{N} \frac{\partial f_c(x_j)}{\partial x_j} (x_{j+1} - x_j) \\ &= \sum_{i=0}^{n} I_i(x) = \int_0^1 \frac{\partial f_c(\gamma(\alpha))}{\partial \gamma(\alpha)} \frac{\partial \gamma(\alpha)}{\partial \alpha} d\alpha \end{aligned} \tag{6}$$

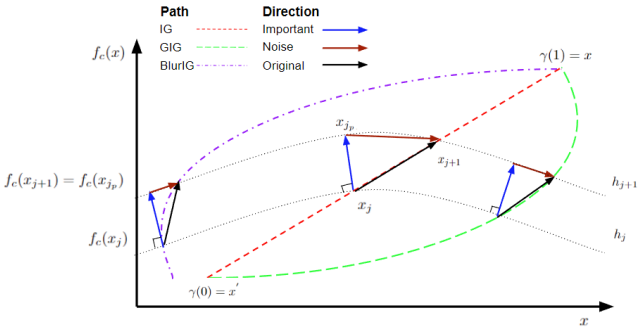

Figure 2. Illustration of IDGI. The red, green, and purple lines illustrate the path corresponding to IG, GIG, and BlurIG, respectively, where both IG and GIG need a reference point as manual input but BlurIG does not. Riemann Integration computes the integral by multiplying the blue and **black** vectors at each calculation step j. 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 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., $\frac{g}{|g|}$, 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 $\frac{g}{|g|}$ 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 h 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 $\frac{g}{|g|}$. 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 $f_c(x) : R^n \to R$, 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 $\overrightarrow{F}$ and further define the hyperplane $h_j = \{x : f_c(x) = f_c(x_j)\}$ in $\overrightarrow{F}$. Assume the Riemann Integration accurately estimates the line integral of the vector field $\overrightarrow{F}$ from points x_j to x_{j+1} and x_{j_p} e.g. $\int_{x_j}^{x_{j_p}} \frac{\partial f_c(x)}{\partial x} dx \approx \frac{\partial f_c(x_j)}{\partial x_j}(x_{j_p} - x_j)$, and $x_j \in h_j$, $x_{j_p}, x_{j+1} \in h_{j+1}$. 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

Algorithm 1 Important Direction Gradient Integration

Inputs: x, f, c, $path : [x', \ldots, x_j, \ldots, x]$

$I_i^{IDGI} = 0$
for $x_j \in path$ **do**
  $d = f_c(x_{j+1}) - f_c(x_j)$
  $g = \frac{\partial f_c(x_j)}{\partial x}$
  $I_i^{IDGI} \mathrel{+}= \frac{g_i \times g_i \times d}{g \cdot g}$
end for
return I^{IDGI}

noise direction. Therefore, the integration at step j consists of two parts: integration from x_j to x_{j_p} and from x_{j_p} to x_{j+1}. While (i) integrating from x_j to x_{j_p} (*important direction*) and then x_{j_p} to x_{j+1} (*noise direction*) and (ii) integrating directly from x_j to x_{j+1} (*original direction*) often assign different attribution values to the features, the target predicted score $f_c(x_{j+1}) - f_c(x_j)$ to be explained at step j stays the same regardless of which path is chosen.

Since $f_c(x_{j_p}) \approx f_c(x_{j+1})$ (as we demonstrated in Theorem 1), the integration of the first path, i.e., from x_j to x_{j_p}, 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_p} 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 $path : [x', \ldots, x_j, \ldots, x]$ 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 $f_c(x_{j+1}) - f_c(x_j)$. In other words, the projection of x_j onto the hyperplane h_{j+1}, formed as $x_{j_p} = x_j + \frac{g}{|g|} \cdot \frac{f_c(x_{j+1}) - f_c(x_j)}{|g|}$, 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.

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 *Senstivity-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+1}, and the two points differ only at i^{th} feature, e.g. $x_j^q \neq x_{j+1}^q, \forall q = i$ and $x_j^q = x_{j+1}^q, \forall q \neq i$. 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 $g_i \times g_i \times (f_c(x_{j+1}) - f_c(x_j)) \neq 0$. 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.

[2] https://github.com/PAIR-code/saliency

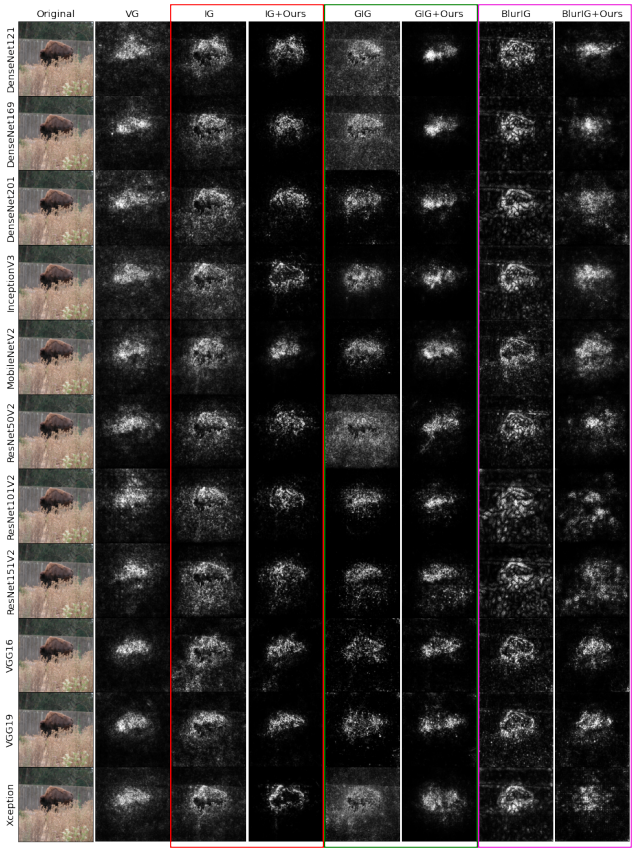

Figure 3. Saliency map comparisons between different methods and different models (each row). Each model correctly predicts the image as *bison*. Left-to-right: Original image, VG, IG, IG+IDGI, GIG, GIG+IDGI, BlurIG, and BlurIG+IDGI. Methods with IDGI focus more on the region of the *bison*.

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-

Metrics	Models	IG-based Methods						Other
		IG	+Ours	GIG	+Ours	BlurIG	+Ours	VG
Insertion Score with Probability (↑)	*DenseNet121*	.293	**.441**	.304	**.364**	.250	**.357**	.176
	DenseNet169	.350	**.503**	.362	**.421**	.298	**.412**	.203
	DenseNet201	.329	**.457**	.341	**.397**	.283	**.395**	.204
	InceptionV3	.348	**.478**	.368	**.460**	.305	**.446**	.214
	MobileNetV2	.203	**.346**	.231	**.291**	.198	**.306**	.129
	ResNet50V2	.338	**.431**	.364	**.403**	.256	**.380**	.219
	ResNet101V2	.377	**.459**	.386	**.412**	.290	**.400**	.248
	ResNet151V2	.388	**.480**	.388	**.418**	.294	**.411**	.242
	VGG16	.217	**.338**	.212	**.265**	.192	**.298**	.134
	VGG19	.232	**.342**	.224	**.273**	.206	**.306**	.145
	Xception	.334	**.487**	.367	**.475**	.298	**.453**	.215
Insertion Score with Probability Ratio (↑)	*DenseNet121*	.322	**.484**	.337	**.400**	.276	**.392**	.194
	DenseNet169	.360	**.518**	.374	**.434**	.307	**.424**	.210
	DenseNet201	.354	**.492**	.370	**.428**	.307	**.425**	.222
	InceptionV3	.389	**.532**	.413	**.512**	.342	**.496**	.240
	MobileNetV2	.248	**.419**	.285	**.354**	.244	**.369**	.160
	ResNet50V2	.353	**.450**	.382	**.422**	.268	**.397**	.229
	ResNet101V2	.389	**.473**	.399	**.425**	.300	**.412**	.256
	ResNet151V2	.399	**.494**	.400	**.430**	.303	**.423**	.250
	VGG16	.248	**.384**	.244	**.303**	.221	**.339**	.155
	VGG19	.265	**.388**	.257	**.313**	.237	**.348**	.167
	Xception	.387	**.564**	.428	**.551**	.347	**.524**	.251

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.

sertion technique inserts pixels from the best to the lowest attribution scores and generates predictions. The approach generates a curve representing the prediction values as a function of the number of inserted pixels. We also compute the normalized curve where the curves are divided by the predicted class probability of the original image. The area under the curves (AUC) are then defined as the insertion scores. The quality of interpretation improves as the insertion score increases. For each image, we iteratively insert the next top 5% important pixel values to the black base image based on each 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.

4.4. 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 by comparing the size of the compressed bokeh image to the size of the compressed original image, which is also referred to as the Normalized Entropy. Bokeh images are binned based on the information level. Then, the average accuracy for each bin is calculated. AIC is the curve of those mean accuracies over bins. Further, the probability of bokeh versus the original image is calculated for each image in each bin. SIC is the curve of the median of those values. Areas under the AIC and SIC curves are computed; better explanation methods should have higher values.

Metrics	Models	IG-based Methods						Other
		IG	+Ours	GIG	+Ours	BlurIG	+Ours	VG
AUC AIC (↑)	*DenseNet121*	.161	**.300**	.141	**.252**	.192	**.230**	.087
	DenseNet169	.160	**.288**	.154	**.254**	.181	**.216**	.089
	DenseNet201	.185	**.307**	.182	**.269**	.213	**.246**	.110
	InceptionV3	.203	**.343**	.189	**.338**	.266	**.301**	.127
	MobileNetV2	.098	**.233**	.114	**.204**	.145	**.197**	.068
	ResNet50V2	.162	**.253**	.162	**.248**	.189	**.210**	.108
	ResNet101V2	.177	**.268**	.163	**.253**	.198	**.215**	.116
	ResNet151V2	.186	**.281**	.165	**.258**	.205	**.229**	.112
	VGG16	.145	**.244**	.141	**.199**	.181	**.222**	.108
	VGG19	.153	**.263**	.150	**.219**	.204	**.240**	.117
	Xception	.238	**.404**	.239	**.381**	.309	**.355**	.174
AUC SIC (↑)	*DenseNet121*	.054	**.228**	.036	**.157**	.085	**.134**	.015
	DenseNet169	.052	**.230**	.045	**.170**	.083	**.130**	.016
	DenseNet201	.068	**.241**	.058	**.183**	.109	**.155**	.019
	InceptionV3	.087	**.294**	.061	**.286**	.171	**.232**	.029
	MobileNetV2	.020	**.145**	.023	**.111**	.043	**.103**	.011
	ResNet50V2	.077	**.210**	.067	**.201**	.099	**.158**	.025
	ResNet101V2	.095	**.231**	.070	**.201**	.117	**.165**	.026
	ResNet151V2	.101	**.249**	.065	**.212**	.122	**.177**	.025
	VGG16	.046	**.166**	.039	**.104**	.082	**.141**	.021
	VGG19	.046	**.177**	.041	**.115**	.098	**.151**	.023
	Xception	.119	**.363**	.107	**.336**	.218	**.296**	.054

Table 2. Area under the curve for AIC and SIC with 11 different models. The information level is computed with the compression size ratio to the original image. IDGI shows improvement for all three IG-based methods across all experiment settings.

Implementation Details. For each image, we first create a base image using PIL[3] with Gaussian blur with a radius of 20 pixels. Then we use 25 percentile values, distributed from 0 to 100, as thresholds (k) to determine the important pixels at each threshold level. That is, given a test instance, we construct 25 bokeh images where each of them contains the top k percent important pixel's original value with the rest pixels' value from the base image. Furthermore, we utilize the implementation from [21] to compute the Normalized Entropy and record the model performance on each bokeh image. For information level bin size, we use 100. We report, in Tab. 2, the AUC under AIC and SIC curves across all baselines and our methods. Our method improves all three IG-based methods across all models drastically.

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

[3]https://pillow.readthedocs.io/en/stable/index.html

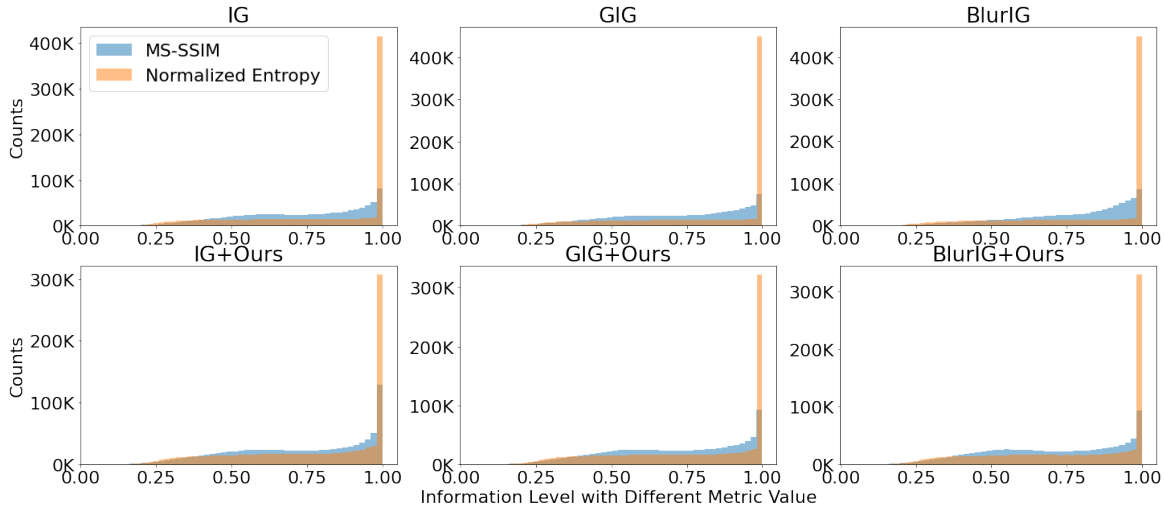

Figure 4. Modified distribution of bokeh images over MS-SSIM and Normalized Entropy [21]. We produce 25 bokeh pictures for each test instance based on the various attribution maps and calculate the MS-SSIM and Normalized Entropy as the information level for each bokeh image. We show the modified bokeh 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.

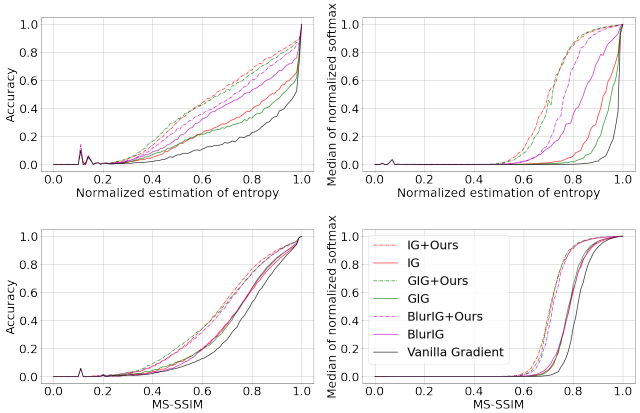

Figure 5. AIC and SIC (median of normalized softmax), from *InceptionV3* model, over the distribution of Normalized Entropy (Top row) and MS-SSIM (Bottom row). The area under the curve indicates IDGI provides significant improvement from all the IG-based methods. As expected, the saliency maps from the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

Metrics	Models	IG-based Methods						Other
		IG	+Ours	GIG	+Ours	BlurIG	+Ours	VG
AUC AIC (↑)	*DenseNet121*	.229	**.305**	.231	**.280**	.216	**.277**	.186
	DenseNet169	.241	**.314**	.249	**.297**	.218	**.289**	.205
	DenseNet201	.254	**.323**	.262	**.303**	.237	**.303**	.216
	InceptionV3	.264	**.333**	.268	**.333**	.264	**.323**	.228
	MobileNetV2	.179	**.259**	.197	**.238**	.186	**.241**	.150
	ResNet50V2	.225	**.277**	.239	**.274**	.209	**.260**	.198
	ResNet101V2	.235	**.284**	.243	**.277**	.215	**.265**	.206
	ResNet151V2	.247	**.302**	.250	**.292**	.227	**.284**	.212
	VGG16	.205	**.271**	.212	**.245**	.204	**.259**	.179
	VGG19	.211	**.275**	.220	**.252**	.214	**.266**	.188
	Xception	.281	**.362**	.293	**.356**	.284	**.345**	.254
AUC SIC (↑)	*DenseNet121*	.184	**.263**	.188	**.239**	.172	**.236**	.139
	DenseNet169	.205	**.282**	.214	**.263**	.182	**.256**	.166
	DenseNet201	.212	**.286**	.221	**.265**	.194	**.266**	.170
	InceptionV3	.211	**.287**	.215	**.285**	.214	**.276**	.179
	MobileNetV2	.126	**.204**	.144	**.187**	.130	**.188**	.096
	ResNet50V2	.196	**.254**	.213	**.250**	.177	**.236**	.167
	ResNet101V2	.210	**.265**	.221	**.256**	.188	**.244**	.180
	ResNet151V2	.221	**.282**	.227	**.270**	.197	**.261**	.186
	VGG16	.163	**.234**	.174	**.210**	.166	**.224**	.137
	VGG19	.173	**.240**	.186	**.219**	.177	**.233**	.149
	Xception	.223	**.312**	.233	**.304**	.229	**.293**	.194

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.

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

Metrics	Models	IG-based Methods						Other
		IG	+Ours	GIG	+Ours	BlurIG	+Ours	VG
F1 (↑)	*DenseNet121*	.663	**.733**	.676	**.700**	.664	**.694**	.648
	DenseNet169	.666	**.730**	.679	**.702**	.667	**.695**	.649
	DenseNet201	.658	**.723**	.675	**.701**	.661	**.694**	.645
	InceptionV3	.661	**.731**	.679	**.727**	.670	**.717**	.651
	MobileNetV2	.666	**.741**	.686	**.713**	.673	**.718**	.656
	ResNet50V2	.673	**.724**	.694	**.720**	.676	**.704**	.668
	ResNet101V2	.675	**.730**	.691	**.716**	.676	**.703**	.666
	ResNet151V2	.675	**.730**	.687	**.713**	.675	**.701**	.663
	VGG16	.672	**.719**	.671	**.696**	.672	**.704**	.667
	VGG19	.672	**.719**	.671	**.696**	.671	**.702**	.666
	Xception	.669	**.745**	.691	**.740**	.678	**.730**	.662
ROC AUC (↑)	*DenseNet121*	.662	**.798**	.660	**.722**	.661	**.722**	.607
	DenseNet169	.656	**.790**	.655	**.714**	.657	**.718**	.582
	DenseNet201	.654	**.788**	.664	**.729**	.658	**.726**	.604
	InceptionV3	.679	**.811**	.664	**.799**	.696	**.790**	.659
	MobileNetV2	.677	**.811**	.695	**.760**	.684	**.778**	.653
	ResNet50V2	.698	**.798**	.698	**.775**	.690	**.746**	.671
	ResNet101V2	.709	**.811**	.693	**.773**	.695	**.753**	.670
	ResNet151V2	.707	**.810**	.681	**.766**	.687	**.745**	.659
	VGG16	.652	**.757**	.648	**.702**	.656	**.727**	.642
	VGG19	.652	**.755**	.650	**.703**	.652	**.719**	.640
	Xception	.693	**.825**	.702	**.814**	.705	**.805**	.683
MAE (↓)	*DenseNet121*	.236	**.189**	.232	**.215**	.238	**.218**	.251
	DenseNet169	.239	**.195**	.235	**.218**	.241	**.222**	.257
	DenseNet201	.237	**.194**	.230	**.212**	.238	**.216**	.251
	InceptionV3	.233	**.187**	.228	**.188**	.227	**.194**	.241
	MobileNetV2	.234	**.183**	.225	**.203**	.231	**.199**	.242
	ResNet50V2	.233	**.195**	.223	**.200**	.234	**.213**	.240
	ResNet101V2	.229	**.189**	.223	**.201**	.232	**.212**	.240
	ResNet151V2	.230	**.189**	.227	**.204**	.234	**.214**	.243
	VGG16	.239	**.206**	.241	**.225**	.240	**.217**	.244
	VGG19	.239	**.206**	.241	**.224**	.241	**.219**	.245
	Xception	.227	**.178**	.218	**.179**	.222	**.186**	.233

Table 4. F1, ROC-AUC, and MAE scores. IG-based methods are improved by IDGI across all three metrics and eleven models.

summarizes the findings for each metric where IDGI consistently outperforms the underlying IG-based approaches.

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 $\otimes$ 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

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 5

[2] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018. 8

[3] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017. 4

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 1, 8

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[6] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015. 1

[7] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 1

[8] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 5

[9] Runmin Cong, Jianjun Lei, Huazhu Fu, Ming-Ming Cheng, Weisi Lin, and Qingming Huang. Review of visual saliency detection with comprehensive information. *IEEE Transactions on circuits and Systems for Video Technology*, 29(10):2941–2959, 2018. 5, 7

[10] Jorge Cuadros and George Bresnick. Eyepacs: an adaptable telemedicine system for diabetic retinopathy screening. *Journal of diabetes science and technology*, 3(3):509–516, 2009. 1

[11] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *Advances in neural information processing systems*, 30, 2017. 1, 8

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[13] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017. 1, 8

[14] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962. 8

[15] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3681–3688, 2019. 8

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 5

[18] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 5

[19] Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. Fastshap: Real-time shapley value estimation. In *International Conference on Learning Representations*, 2021. 1, 8

[20] Parimala Kancharla and Sumohana S. Channappayya. Improving the visual quality of generative adversarial network (gan)-generated images using the multi-scale structural similarity index. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3908–3912, 2018. 6

[21] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. Xrai: Better attributions through regions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4948–4957, 2019. 1, 5, 6, 7, 8

[22] Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5050–5058, 2021. 1, 2, 4, 5, 8

[23] Saeed Khorram, Tyler Lawson, and Li Fuxin. igos++ integrated gradient optimized saliency by bilateral perturbations. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 174–182, 2021. 8

[24] Ping Liu and Mustafa Bilgic. Relational classification of biological cells in microscopy images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 344–352, 2021. 1

[25] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. 1, 8

[26] Kede Ma, Qingbo Wu, Zhou Wang, Zhengfang Duanmu, Hongwei Yong, Hongliang Li, and Lei Zhang. Group mad competition? a new methodology to compare objective image quality models. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1664–1673, 2016. 6

[27] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019. 1, 8

[28] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017. 6

[29] Deng Pan, Xin Li, and Dongxiao Zhu. Explaining deep neural network models with adversarial gradient integration. In *IJCAI*, pages 2876–2883, 2021. 1, 5, 8

[30] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018. 1, 5, 8

[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

[32] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 1, 8

[33] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 1, 8

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

[35] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017. 1, 8

[36] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 1, 5, 8

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[38] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 1, 8

[39] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 1, 8

[40] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014. 1, 8

[41] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 2020. https://distill.pub/2020/attribution-baselines. 1

[42] Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR, 2020. 1, 8

[43] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017. 1, 2, 4, 5, 8

[44] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 5

[45] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003. 6

[46] Shawn Xu, Subhashini Venugopalan, and Mukund Sundararajan. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9680–9689, 2020. 1, 2, 5, 7, 8

[47] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 1, 8

[48] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*, 2017. 1, 8