Using the Path of Least Resistance to Explain Deep Networks

Anonymous Authors[1]

Abstract

Integrated Gradients (IG), a widely used path-based attribution method, assigns importance scores to input features by integrating gradients of the models along a straight path from a baseline to the input. While effective in certain cases, we show that choosing straight paths can lead to flawed attributions. In this paper, we identify how these misattributions arise. As a solution, we propose a new approach that considers the input space as a Riemannian manifold and computes attributions by integrating gradients of the model along geodesics. We call our approach *Geodesic Integrated Gradients*. To approximate geodesic paths, we introduce two methods: a *k*-nearest neighbour-based approach for simpler models and a Stochastic Variational Inference-based method for more complex ones. Furthermore, in our experiments with both synthetic and real world data, we demonstrate that our approach outperforms existing explainability methods including the original IG.

1. Introduction

The use of deep learning models has risen in many applications. With it, so too has the desire to understand why these models make certain predictions. These models are often referred to as "opaque", as it is difficult to discern the reasoning behind their predictions (Marcus, 2018). Additionally, deep learning models can inadvertently learn and perpetuate biases found in their training data (Sap et al., 2019). To create fair and trustworthy algorithms, it is essential to be able to explain a model's output (Das & Rad, 2020).

Some examples of the methods proposed to explain neural networks include Gradient SHAP (Lundberg & Lee, 2017), Integrated Gradients (Sundararajan et al., 2017) and (Kapishnikov et al., 2021).

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

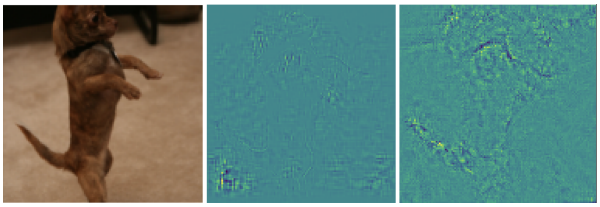

Figure 1. Comparing attributions of Integrated Gradients (middle image) with Geodesic Integrated Gradient (right image). In this work we show how to generate shaper explanations by integrating gradients along the geodesic line, as opposed to the straight line.

Significant effort has been dedicated to designing explanation methods that satisfy certain desirable axioms. This is due to the lack of ground truth for evaluating them. The axioms can ensure that the explanations are principled. One of the most successful axiomatic methods is Integrated Gradients (IG) (Sundararajan et al., 2017). Consider a function $f : R^n \to R$, representing the neural network and an input vector $\mathbf{x} \in R^n$. Furthermore, consider a baseline input vector $\overline{\mathbf{x}} \in R^n$ (typically chosen such that the network gives baseline a near zero score). IG explains the network by quantifying how much of the difference $f(\mathbf{x}) - f(\overline{\mathbf{x}})$ can be attributed to the ith dimension of $\mathbf{x}$, $\mathbf{x}_i$.

Integrated Gradient gives attribution IG_i to the ith dimension of the input by solving the following path integral

$$IG_i(\mathbf{x}) = (\mathbf{x}_i - \overline{\mathbf{x}}_i) \int_0^1 \frac{\partial f(\gamma(t))}{\partial \mathbf{x}_i} dt, \qquad (1)$$

where $\gamma(t) = \overline{\mathbf{x}} + t(\mathbf{x} - \overline{\mathbf{x}})$ is a straight path from the baseline to input. The claim of the creators of IG is that Eq. 1 tells us how the model got from predicting essentially nothing at $\overline{\mathbf{x}}$ to giving the prediction at $\mathbf{x}$. Considering gradients represent the rate of change of functions, the above expression should tell us how scaling each feature along the path affects the increase in the network score for the predicted class.

In this paper, we demonstrate that defining attributions along straight paths in Euclidean space can lead to misattributions. We examine the consequences of these issues through examples in computer vision, as shown in Fig. 1, alongside simpler illustrative cases in Fig. 2. To address these challenges, we introduce **Geodesic Integrated Gradients** [1], a gener-

[1]The code for the attribution methods and reproducing the

alisation of IG that replaces straight paths with geodesic ones. These geodesics are defined on a Riemannian manifold, characterised by the model's input space and a metric induced by the model's gradients. This approach mitigates the identified pitfalls while retaining all the axioms of IG.

Before making the case for our Geodesic Integrated Gradient, let us first show an example of an artefact that can arise from choosing straight paths, generating explanations which do not reflect the true behaviour of a model.

We highlight this issue on a half-moons classification task. We train a simple multi-layer perceptron (MLP) with 3 layers, ReLU activations and a cross-entropy loss to distinguish the upper moon from the lower one. The cross-entropy is split into a final log-softmax activation and a negative log-likelihood loss, so that we can explain probabilities.

We now compute Integrated Gradients, Eq. 1, for this model on the test data. Let us consider a baseline and input pair, such that the baseline is outside of either half-moon, for example at (-0.5, -0.5). This is a good choice of baseline, since network should assign near-zero score to it. Let us call the feature in the vertical axis the 1st component of x. In Fig. 2 we illustrate the attribution of this feature, $IG_1(\mathbf{x})$, for each point using the colour map. One should expect to see all the points sufficiently above the decision boundary to receive equally high attributions. Intuitively, this is expected, because if a point is above the decision boundary, its x_1 component is an important factor in the classification. However, for a model that is very skillful at the classification task, since the point is significantly above the decision boundary, going slightly down should not make any difference. Because in such a model's score should not change significantly anywhere other than near the decision boundary. However, we can see in Fig. 2 that some points on the upper moon receive much higher x_1-attribution than others. These are points such that a straight line from the baseline to them mostly falls on high gradient regions. This does not reflect the model's behaviour. A similar point could be made about the horizontal axis. This is in contrast with Fig. 2, where we show our method gives equally high attribution to all points sufficiently above the decision boundary, with different shades for the points closer to the boundary in x_1 direction. In Section 3.1 we present the details of how our method that achieves the results presented in this figure.

Before giving the formal method in section 2, let us discuss the intuition behind our Geodesic IG. We want the path in Eq. 1 to be such that it avoids regions with high model gradient. This is because failing to do so would superficially increase the result of the integral, leading to the types of

experiments is provided on https://github.com/sina-salek/geodesic-ig

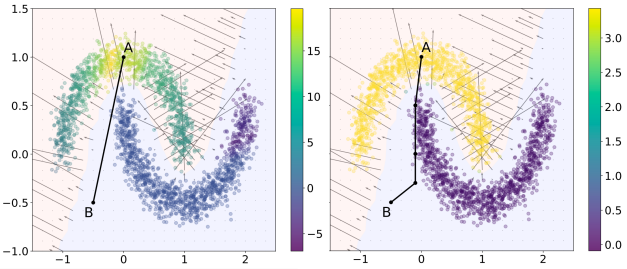

Figure 2. **Integrated Gradients attributions.** The colour map represents $\mathrm{IG}_1(\mathbf{x})$ for each point $\mathbf{x}$, with $(-0.5, -0.5)$ (point B) as the baseline. Points around A have much higher attributions than other points on the top moon, despite all being sufficiently above the decision boundary of the MLP. This is due to the path being close to the decision boundary, resulting in high gradients (gray arrows) along this path.

artefacts illustrated in Fig. 2. Therefore, we should try to find the path of least resistance, as this is the path that avoids steep gradients as much as possible. As we shall see in section 2, the input space can be viewed as a Riemannian manifold with a metric derived from the model gradients.

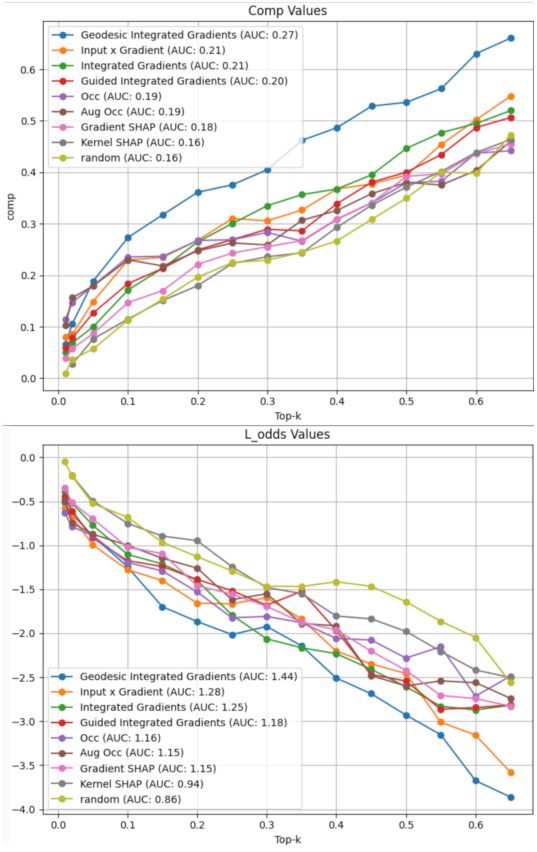

Figure 3. **Metrics comparison** Evaluation of our results via comprehensiveness (DeYoung et al., 2019) and log odds (Shrikumar et al., 2017) show Geodesic Integrated Gradients significantly outperforming other methods.

The path of least resistance between a chosen baseline and input, therefore, is the geodesic path between the two points.

In section 2, we develop two methods to approximate the geodesic path between two points on the manifold. We use the first method, kNN-based for simpler manifolds, and the second one, based on Stochastic Variational Inference, for more complex manifolds. We then show that geodesic IG satisfies all the axioms of IG.

In Section 3, we demonstrate the effectiveness of the Geodesic IG method on the real-world Pascal VOC 2012 dataset (Everingham et al.). Our results outperform existing methods, as we evaluate using various metrics. We preview the results of this experiment in Fig. 3

Section 4 reviews related work, including the comparison of Geodesic IG with other methods that attempt to overcome the shortcomings of Integrated Gradients.

2. Method

In section 1, we gave the intuition that using geodesic paths can correct the misattribuion in IG that arise from integrating along straight paths. Let us now formalise this idea.

2.1. Geodesic distance formulation.

Let us define a neural network as a function $f : \mathbb{R}^n \to \mathbb{R}$, where n is the dimension of the input space. Let us also define $\mathbf{x}$ a point in this input space. We denote the Jacobian of f at $\mathbf{x}$ as $\mathrm{J}_\mathbf{x}$.

Using Taylor's theorem, for a vector $\boldsymbol{\delta}$ with an infinitesimal norm: $\forall \epsilon, ||\boldsymbol{\delta}|| \leq \epsilon$, we have:

$$||f(\mathbf{x}+\boldsymbol{\delta}) - f(\mathbf{x})|| \approx ||\mathrm{J}_\mathbf{x}\boldsymbol{\delta}|| \approx \boldsymbol{\delta}^T \mathrm{J}_\mathbf{x}^T \mathrm{J}_\mathbf{x} \boldsymbol{\delta} \tag{2}$$

Using equation 2, we can now define a tangent space $\mathrm{T}_\mathbf{x}\mathrm{M}$ of all $\boldsymbol{\delta}$, equipped with a local inner product $\mathrm{G}_\mathbf{x}$:

$$< \boldsymbol{\delta}, \boldsymbol{\delta}' >_\mathbf{x} = \boldsymbol{\delta}^T \mathrm{G}_\mathbf{x} \boldsymbol{\delta}' = \boldsymbol{\delta}^T \mathrm{J}_\mathbf{x}^T \mathrm{J}_\mathbf{x} \boldsymbol{\delta}' \tag{3}$$

As a result, we can view the input space as a Riemannian manifold $(\mathbb{R}^n, \mathrm{G})$, where the Riemannian metric G is defined above. On this manifold, the length of a curve $\gamma(t) : [0,1] \to \mathbb{R}^n$ is defined as:

$$\begin{aligned} \mathrm{L}(\gamma) &= \int_0^1 \sqrt{< \dot\gamma(t), \dot\gamma(t) >_{\gamma(t)}} dt \\ &= \int_0^1 ||\partial_t f(\gamma(t)) \times \dot\gamma(t)||\, dt, \end{aligned} \tag{4}$$

where $\dot\gamma(t)$ is the derivative of $\gamma(t)$ with respect to t. The **geodesic distance**, denoted L^*, between **a** and **b** is then defined as the minimum length among curves γ such that $\gamma(0) = \mathbf{a}$ and $\gamma(1) = \mathbf{b}$. We also call **geodesic path** the curve γ^* which minimises the length L. This path can be interpreted as the shortest path between **a** and **b** in the manifold.

Remark 2.1. We can infer from Equation 4 that the geodesic path avoids as much as possible high-gradients regions. This is the main desired property of a path to be used for path-based attributions. Representing the path of least resistance, the geodesic path circumvents superficially high values of attributions.

2.2. Approximation of the geodesic with K Nearest Neighbours.

Computing the exact geodesic would require computing L on an infinite number of paths γ, which is not possible in practice. However, several methods have been proposed to approximate this value. We draw from previous work (Yang et al., 2018; Chen et al., 2019) and present one with desirable characteristics.

First, we compute the K Nearest Neighbours (kNN) algorithm on points between (and including) input and baseline. These points can be either sampled or generated. The geodesic distance between two neighbouring points, $\mathbf{x}_i$ and $\mathbf{x}_j$, can be approximated by a straight path $\mathbf{x}_i + t \times (\mathbf{x}_j - \mathbf{x}_i)$. We have the above approximation because for dense enough data, the euclidean distance between neighbouring points is a good approximation of the geodesic distance. This reflects the fact that a small region of a Riemannian manifold, called Riemann neighbourhood, is locally isometric to a Euclidean space[2]. So the geodesic distance between the two neighbouring points is approximated by:

$$\begin{aligned} \mathrm{L}^*_{ij} &= \int_0^1 ||\partial_t f(\mathbf{x}_i + t \times (\mathbf{x}_j - \mathbf{x}_i)) \times (\mathbf{x}_i - \mathbf{x}_j)||\, dt \\ &= ||\mathbf{x}_i - \mathbf{x}_j|| \int_0^1 ||\partial_t f(\mathbf{x}_i + t \times (\mathbf{x}_j - \mathbf{x}_i))||\, dt \end{aligned} \tag{5}$$

Equation 5 corresponds to the original Integrated Gradients method, albeit with the norm. This integral can be approximated by a Riemannian sum similarly to (Sundararajan et al., 2017):

$$\mathrm{L}^*_{ij} \approx ||\mathbf{x}_i - \mathbf{x}_j|| \sum_{k=0}^{m} ||\partial f(\mathbf{x}_i + \frac{k}{m} \times (\mathbf{x}_j - \mathbf{x}_i))|| \tag{6}$$

For input-baseline pair, $\mathbf{x}$ and $\overline{\mathbf{x}}$, we can now see the set

[2]We shall further formalise this intuition later in this section.

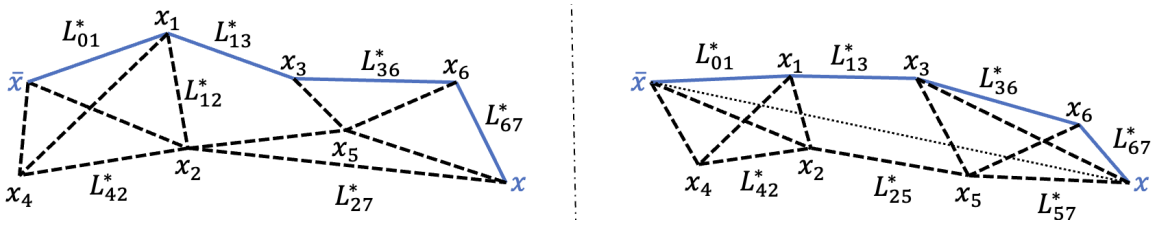

Figure 4. **Method overview.** For an input $\mathbf{x}$, a baseline $\bar{\mathbf{x}}$, and a set of points $\mathbf{x}_i$, we compute the kNN graph using the euclidean distance (dashed lines). For each couple $(\mathbf{x}_i, \mathbf{x}_j)$, we then compute the integrated gradients L^*_{ij} using Equation 6. For clarity, not all L^*_{ij} are present on the figure. 0 and 7 represent $\bar{\mathbf{x}}$ and $\mathbf{x}$ respectively. Using the resulting undirected weighted graph, we use the Dijkstra algorithm to find the shortest path between $\mathbf{x}$ and $\bar{\mathbf{x}}$ (blue continuous lines). On the left, the points $\mathbf{x}_i$ are provided while, on the right, the points are generated along the straight line between $\mathbf{x}$ and $\bar{\mathbf{x}}$ (dotted line).

$(\mathbf{x}, \bar{\mathbf{x}}, \mathbf{x}_i)$ as a weighted graph, with the weights being the geodesic distances between two neighbors L^*_{ij}. To compute the geodesic path between $\mathbf{x}$ and $\bar{\mathbf{x}}$, we can use a shortest path algorithm, such as Dijkstra or A* with the euclidean distance as the heuristic.

The resulting Geodesic Integrated Gradients corresponds to the sum of the gradients along this shortest path:

$$\text{Geodesic IG}_i(\mathbf{x}) = (x_i - \bar{x}_i) \sum_{k=0}^{m} \int_0^1 \frac{\partial f(\mathbf{x}^k + t \times (\mathbf{x}^{k+1} - \mathbf{x}_k))}{x_i^k} \, dt \tag{7}$$

where $\mathbf{x}^k$ are the points along the shortest path. The integrals in Equation 7 can also be approximated with Riemannian sums.

The gradients between each pair of neighbours can also be estimated in batches to speed up the attribution computation. Moreover, several inputs' attributions can be computed together, with similar speed as IG: if we want to compute the attribution of N inputs, with 10 interpolation steps and 5 nearest neighbors, the number of gradients to calculate is $10 \times 5 \times \mathrm{N} = 50\mathrm{N}$, which amounts to computing IG with 50 steps. This does not include the computation of the shortest path, which is for instance $O(\mathrm{N}^2)$ for Dijkstra algorithm. Please also refer to Figure 4 for an illustration of this method.

Assumption of the approximation. Here we formalise the intuition that, for a pair of neighbours, the geodesic path between them is close to the euclidean one. Notice that the derivative of the neural network f is Lipschitz continuous,

$$\exists \mathrm{K} \, \forall \mathbf{x}, \mathbf{y}, \; ||\mathrm{J}_\mathbf{x} - \mathrm{J}_\mathbf{y}|| \leq \mathrm{K} \times ||\mathbf{x} - \mathbf{y}||. \tag{8}$$

Equation 8 is equivalent to the Hessian of f being bounded. Under this assumption, if two points $\mathbf{x}$ and $\mathbf{y}$ are close enough, the Jacobian of one point is approximately equal to the other: if $||\mathbf{x} - \mathbf{y}|| \leq \epsilon$, then $\mathrm{J}_\mathbf{x} \approx \mathrm{J}_\mathbf{y}$. As a result, the length between $\mathbf{x}$ and $\mathbf{y}$, for a curve γ, is: $\mathrm{L}(\gamma) \approx \int_\gamma ||\mathrm{J}_\mathbf{x}|| \, d\mathbf{x} \approx ||\mathrm{J}_\mathbf{x}|| \int_\gamma d\mathbf{x}$. Due to the triangular inequality, the shortest path γ^* is then a straight line, and we have: $\mathrm{L}^*(\mathbf{x}, \mathbf{y}) \approx ||\mathrm{J}_\mathbf{x}|| \times ||\mathbf{x} - \mathbf{y}||$.

As a result, under this assumption, if two points are close, the geodesic path can be approximated with a straight line. Note that even though we take the path between two neighbouring points to be a straight line, we do not assume that the Jacobian of the function between the two points is constant.

Handling disconnected graphs An issue with the graph computed with the kNN algorithm is that it could be disconnected, in which case it could be impossible to compute a path between an input and a baseline. To alleviate this issue, we add so called "bridges" to the graph, as following: for each disconnected component, we add one link between them, specifically between two points of each component having the lowest euclidean distance. An illustration of this method is displayed on Figure 5.

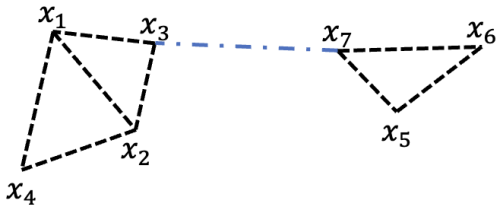

Figure 5. When the kNN graph is disconnected, as illustrated here, it would be impossible to compute Geodesic IG between certain points, for instance $\mathbf{x}_1$ and $\mathbf{x}_5$ here. To solve this, we add a single link between disconnected graphs, here between $\mathbf{x}_3$ and $\mathbf{x}_7$.

However, we stress that this solution is not optimal, and argue that a better way of handling this issue would be to

avoid disconnected graphs in the first place. This can be done by increasing the number of neighbours k.

2.3. Approximation of the geodesic with energy-based sampling.

While our kNN-based method is effective for explaining simpler models, its applicability diminishes as model complexity increases. In such cases, a prohibitively large number of samples is required between the baseline and the input to provide accurate estimates of the geodesic path. Even with relatively large number of samples, it is not trivial where on the manifold to sample the points to adequately capture the gradient landscape. Furthermore, once the points are sampled, searching the graph for the shortest path will be computationally too intensive. For such use-cases, we devise another approximation procedure based on an energy-base sampling method, such as Stochastic Variational Inference (SVI).

As we observed in Remark 2.1, we would like to sample from the shortest paths between two points on the input space that avoids high gradient regions as much as possible. Therefore, we want to deviate from the straight line to avoid high gradient regions. This process can be approximated in the following way. First we start with a straight line between the two points and define a potential energy as a combination of two terms. One for the distance of the path to be optimised to the straight line, and the other is a curvature penalty term. Finding the minimum energy path, therefore samples approximately on the geodesic lines. More formally, let us define the distance term: $d(x, y) := |x - y|_2$, and the curvature term: $c(x) := |\nabla f(x)|_2$ where f is the neural network. The total energy being minimised is

$$E(\gamma) = \sum_{i=1}^{n} \|\gamma_i - \gamma_i^0\|_2 - \beta \|\nabla f(\gamma_i)\|_2, \qquad (9)$$

where γ is the path γ^0 is the initial path $beta$ controls trade-off between distance and curvature.

With this energy function, one can use a suitable sampling method, such as SVI or Hamiltoniam Monte Carlo to sample points on the geodesic paths. Here we briefly describe the SVI optimisation, as this has a good balance of computational efficiency and accuracy.

SVI provides a probabilistic framework for optimising paths between input and baseline points. To achieve this, SVI defines a probability distribution $p(\gamma|\gamma_0)$ proportional to $exp(-E(\gamma))$, where $E(\gamma)$ is our defined potential energy. Rather than directly sampling from this complex distribution, we introduce a simpler variational distribution $q(\gamma)$ parameterised by learnable means and scales. This guide distribution takes the form of a factorised normal distribution $\prod_i N(\mu_i, \sigma_i)$ over path deviations.

The optimisation proceeds by minimising the KL divergence between $q(\gamma)$ and the true posterior through maximisation of the Evidence Lower Bound (ELBO). Critically, this allows us to learn optimal parameters for $q(\gamma)$ through gradient-based optimisation. The learned means μ_i define the optimal path deviations from the initial straight-line path, while the scales σ capture uncertainty in these deviations. This probabilistic approach naturally samples of the low-energy regions.

We use this method in our computer vision experiments and show its efficacy in section . However here we find it instructive to visualise these paths on the 2D example of half-moons, Fig. 6, even though for simpler use-cases such as this we would prefer the kNN method, as they are easier to control.

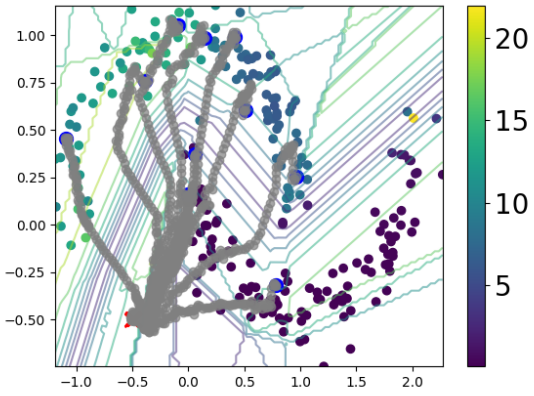

Figure 6. **Visualising 10 random paths.** For a simple case of half-moons with 1000 samples, we show the sampled paths between 10 pairs of points. As we see on low gradient regions the sampler prefers straigh lines, whereas in high gredient regions the path becomes closer to perpendicular to the large gradient vectors to cross the region as quickly as possible.

2.4. Axiomatic properties

The family of generalisations of Integrated Gradients to non-straight paths, such as Eq. 7, is called *path methods* of explanation. We see in Sundararajan et al. (2017) that all path methods satisfy all of the axioms that IG is based on, apart from the symmetry axiom. Therefore, here we focus on this axiom only.

Symmetry preserving of Geodesic IG The symmetry axiom is defined in the following way.

Definition 2.2. Consider an input-baseline pair $\mathbf{x}$ and $\overline{\mathbf{x}}$,

and a function f that is symmetric in dimensions i and j. If $\mathbf{x}_i = \mathbf{x}_j$ and $\overline{\mathbf{x}}_i = \overline{\mathbf{x}}_j$, then an attribution method is Symmetry-Preserving if $attr_i(\mathbf{x}; f) = attr_j(\mathbf{x}; f)$, where $attr_n(\mathbf{x}; f)$ is the attribution of $\mathbf{x}_n$.

(Sundararajan et al., 2017, Theorem 1) shows that IG is the only path method that satisfies symmetry for any function. However, as noted in (Kapishnikov et al., 2021), while straight path is the only one satisfying symmetry for any function, given a specific function, it may be possible to find other paths that satisfy Symmetry. We generalise this theorem for Geodesic IG on Riemannian manifolds and how that our method satisfies Symetry for the neural network function we sample the paths for.

Here we demonstrate that Geodesic Integrated Gradients satisfies symmetry property on Riemannian manifolds induced by the input space and the gradients of the model.

Let the ith and jth dimensions of $\gamma(t)$ be $\gamma_i(t)$ and $\gamma_j(t)$ respectively and f be a function differentiable almost everywhere on t. Furthermore, take f to be symmetric with respect to x_i and x_j. If $\gamma_i(t) = \gamma_j(t)$ for all $t \in [0, 1]$, then we have

$$||\partial_t f(\gamma_i(t)) \times \dot{\gamma}_i(t)|| = ||\partial_t f(\gamma_j(t)) \times \dot{\gamma}_j(t)||, \quad (10)$$

almost everywhere on t. Therefore, the ith and jth components of Eq. 4 are equal. Furthermore, since Eq. 7 integrates along the path that is an approximation of Eq. 4, we have Geodesic IG_i = Geodesic IG_j. Indeed our geodesic paths satisfy $\gamma_i(t) = \gamma_j(t)$ for all $t \in [0, 1]$ on the Riemannian manifolds. To see this, let us select a baseline $\overline{\mathbf{x}}$ and U a Riemann neighbourhood centred at $\overline{\mathbf{x}}$. Let us also define the geodesic path γ such as $\gamma(0) = \overline{\mathbf{x}}$. Further, define $\mathbf{v}(t) := \gamma'(t)$, where γ' is the derivative of γ. Then, in the local coordinates system of the neighbourhood of any point, called normal coordinates, we have $\gamma(t) = (tv_1(t), ..., tv_n(t))$. Since the function is symmetric in the ith and jth dimensions, we have v_i and v_j are the same everywhere. From this, we can see that $\gamma_i(t) = \gamma_j(t)$ for all $t \in [0, 1]$ and therefore Geodesic IG satisfies symmetry.

3. Experiments

To validate our method, we performed experiments on two datasets: one is the synthetic half-moons dataset, and the other is the real-world Pascal VOC 2012 dataset.

3.1. Experiments on the half-moons dataset

Here we give the details of the half-moons experiment discussed in the Introduction section. We use the half-moons dataset provided by Scikit learn (Pedregosa et al., 2011) to generate 10,000 points with a Gaussian noise of $\mathcal{N}(0, 0.2)$. The dataset is split into 8,000 training points and 2,000 testing ones. The model used is an MLP.

We measure two indicators of performance for each attribution method: a lack of artifacts not reflecting the model's behaviour, and a low variation of attributions between close points. To that goal, we use two metrics, **purity** and **standard deviation**, defined in the following way. We know that a well-trained model should classify about half of the data points as upper moon, class 1, and the other half as lower moon, class 0. Such a model should consider both features of each point important for the classification into class 1. Therefore, for such a model, we expect in a good attribution method, the top 50% points as ranked by the quantity $\widetilde{attr}(\mathbf{x}; f) = \sum_{i=0}^{1} |attr_i(\mathbf{x}; f)|$, to be classified as 1 (assuming the baseline is chosen as a point to which the network gives a near-zero score). With this in mind, purity is defined as

$$\text{Purity} = \frac{1}{N/2} \sum_{\mathbf{x},\, \widetilde{attr}(\mathbf{x};f) \in \text{Top } 50\% \text{ of all attr}} \text{argmax}(f(\mathbf{x})), \quad (11)$$

where N is the number of data points. We see that this is the average value of the predicted class labels for half of the points. From the above, we infer that, for a well-trained model, we prefer an attribution method that results in the purity close to 1. In contrast, a random attribution method in this case would result in the purity score of 0.5.

For the second metric, standard deviation, points that belong to the same moon to have similar $\widetilde{attr}(\mathbf{x}; f)$. Therefore, define

$$\text{Std}_i = \text{std}\{\widetilde{attr}(\mathbf{x}; f),\ \text{argmax}(f(\mathbf{x})) \in \text{Moon }_i\}. \quad (12)$$

We expect this standard deviation metric to be low for the points belonging to either class.

In this experiment, we compare the results of attributions from Geodesic IG with the original IG, as well as more recent comparable methods including Enhanced IG (Jha et al., 2020), GradientShap (Lundberg & Lee, 2017), SmoothGrad (Smilkov et al., 2017).

For all of the methods, we use $(-0.5, -0.5)$ as a baseline. Enhanced IG is an improvement over IG where the kNN algorithm is used to avoid computing gradients on paths on out of sample distributions. The chosen number of neighbours for the kNN part of both Enhanced IG and Geodesic IG is 5. We perform an ablation study of this parameter in Appendix A.

The quantitative analysis of the results are given in Table 1, where we see that Geodesic IG significantly outperforms all other methods on all metrics. In particular, notice that all other methods perform relatively poorly on STD_1. This is because for this metric, the integration path has to cross the

Method	Purity ↑	STD_0 ↓	STD_1 ↓
GradientShap	0.761	1.90	5.01
IG	0.802	1.44	4.915
SmoothGrad	0.800	1.42	4.720
Enhanced IG	0.738	1.23	1.876
Geodesic IG	**0.978**	**0.237**	**0.267**

Table 1. Evaluation of different attribution methods on a half-moons dataset with Gaussian noise $\mathcal{N}(0, 0.2)$. The results over 5 different seeds are averaged, with the corresponding standard deviation in brackets. We present in Appendix A more results with different amounts of Gaussian noise.

decision boundary. However, the other methods might cross the decision boundary differently and regardless of the function's gradient, resulting in spuriously different attributions for different points belonging to class 1. In Appendix A we see that the gap between the performance of geodesic IG and the other methods increases as the Gaussian noise of the half-moons increases. To provide better understanding of these results we present more analysis on this dataset in Section 4.

3.2. Experiments on the Pascal VOC 2012 dataset

Now we would like to test our method on a real-world dataset. To this aim, we use the Pascal VOC 2012 dataset, which consists of labelled images (Everingham et al.). We trained a classification head on this dataset and added it to the pre-trained ConvNext model from torch vision to generate predictions to be explained (He et al., 2016). We also perform the analysis on 100 randomly sampled images from the test set.

We compare our method with various baselines: Integrated Gradients, GradientShap, InputXGradients (Shrikumar et al., 2016), KernelShap (Lundberg & Lee, 2017), Occlusion (Zeiler & Fergus, 2014) Augmented Occlusion (Tonekaboni et al., 2020) and Guided IG (Kapishnikov et al., 2021).

Where an explainer needs a baseline, such as in IG as well as our method, a uniformly black image was chosen as baseline.

To evaluate the performance of an attribution method, we use 2 different metrics:

- **Comprehensiveness** (DeYoung et al., 2019): We mask the top k% most important features in absolute value, and compute the average change of the predicted class probability compared with the original image. A higher score is better as it indicates masking these features results in a large change of predictions.
- **Log-odds**[3] (Shrikumar et al., 2017): We mask the top k% most important features in absolute value, and measure the negative logarithmic probabilities on the predicted class compared with the original one. Lower scores are better.

Method	AUC-Comp ↑	AUC-LO ↑
Input X Gradients	0.21	1.28
GradientShap	0.18	1.15
IG	0.21	1.25
Random	0.16	0.86
Kernel Shap	0.16	0.94
Occlusion	0.19	1.16
Aug Occlusion	0.19	1.15
Guided IG	0.20	1.18
Geodesic IG	**0.27**	**1.44**

Table 2. Evaluation of different attribution methods on 100 randomly sampled images from the Pascal VOC test set. Fig. 3 shows the curves where these metrics are extracted from.

We calculated these two metrics for a range of top-k%s, from 1% to 56%, which is shown in Fig. 3. To summarise the values from different top-k%s, we calculate the area between the curves and the horizontal line at $y = 0$, a.k.a. area under the curve, and report them in Table 2. Fig. 1 provides a qualitative comparison between Geodesic IG and Integrated Gradients. The analysis shows that Geodesic IG outperforms other methods in explaining the model's behaviour on the dataset, and it does so with a striking gap in the case of comprehensiveness.

The one downside of using Geodesic IG for explaining complex deep learning models is the computational cost. Running the above experiment on 100 images took 23 hours using an L4 GPU. However, this would be a reasonable price to pay to get more accurate explanations in use-cases where understanding the model is more critical.

4. Related Work

Approximating geodesic paths is a widely studied area of research, and many methods to do so have been developed. For a comprehensive survey on this subject, please refer to Crane et al. (2020). Our work is specifically inspired from the ISOMAP method (Tenenbaum et al., 2000), a dimensionality reduction method which approximates geodesic

[3]This metric should be called *Log-probabilities*. However, since Log-odds is a commonly used name in the literature, we refer to it as Log-odds.

paths on a manifold. However, our method differs from ISOMAP in that we weight our graph not using euclidean distance, but the norms of a model's gradients. Our aim is indeed not to model the input space, but to explain a neural network by building paths avoiding high-gradients regions.

As mentioned in Section 3, the idea of using a kNN algorithm to avoid computing gradients on out of distribution data points has also been used in Enhanced Integrated Gradients Jha et al. (2020). However, this method creates a path which is model agnostic, as it does not necessarily avoid high gradients regions. As a result, it can lead to significant artefacts which do not reflect the model's behaviour. To support this argument, we provide an example where this method fails on the half-moons datasets (Figure 7).

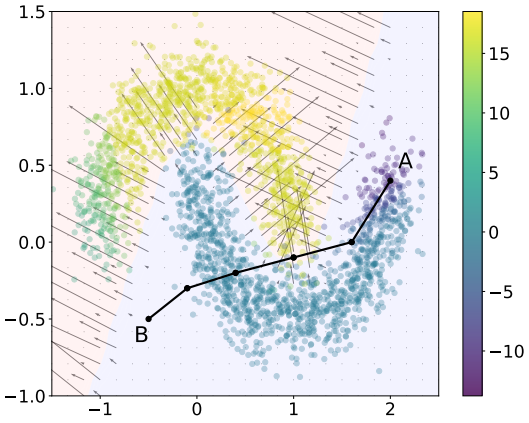

Figure 7. **Enhanced IG attributions.** Enhanced IG computes a kNN algorithm, uses Dijsktra to find the shortest path between an input and a reference baseline, and computes gradients along this path. However, this method is model agnostic and can as a result cross a high gradients region, which is the case in this example, between the input A and the baseline B. Input A therefore has a high attribution which does not reflect the model's true behavior. In this example, the noise is $\mathcal{N}(0, 0.15)$.

The idea of adapting the path according to the model has be proposed by Kapishnikov et al. (2021), calling their method Guided Integrated Gradients. Their method computes this path greedily by selecting around 10% of the features that have the lowest absolute value of the partial derivatives, and switching these features from the baseline's values to the input's values. However, as the authors indicate, such method can create out of distribution data points, with part of their features either matching the input or the baseline. As a result, they add a hyperparameter K, which forces the path to go through K points along the straight line between the input and the baseline. We argue, however, that, by directly approximating the path of least resistance, our method is more principled compared to Guided IG. In Guided IG, it is not clear how to choose the value of K: a too low value could create out of distribution samples, while a too high one would force the path to be close to the straight line and potentially cross high gradients regions. It is similarly not clear why switching 10% of the features, and how to tune this hyperparamer. On the other hand, we argue that Geodesic IG does not have this issue. It has indeed two hyperparamers: the number of points used to approximate the geodesic path, and the number k of nearest neighbours. Yet, the performance of Geodesic IG should improve when both hyperparamers values increase, as, when more points or more neighbours are used, the approximation of the geodesic path should improve. Increasing these values would, however, require more computing and performance needs to be balanced with the amount of compute available.

5. Discussion

We have identified in this paper potential issues with path-based attribution methods: the presence of artefacts due to ignoring the curvatures of the model. To overcome these issues, we have introduced Geodesic Integrated Gradients, an adaptation of the original IG method which integrates gradients not along a straight line, but along the geodesic of a manifold defined by the model.

By avoiding high-gradients regions in the input space, we have shown that Geodesic IG can successfully address these issues. Moreover, it follows all of the axioms defined by Sundararajan et al. (2017).

We have presented two methods to approximate these geodesic paths: one based on kNN and the other Stochastic Variational Inference. Whilst our method shows clear advantage over the alternatives, evaluated by metrics such as Comprehensiveness and Log-Odds, there are some further research questions that we leave for future explorations. One is around computational intensiveness of our methods, as we discussed in section 3. The other is the inherent noise in any sampling process that we may wish to use for approximating the geodesic lines. Regarding the noise, it is concievable that a method involving directly solving the geodesic equation might result in less noisy approximation of the geodesic paths. Depending on the way one solves such equations, it is possible that finding the geodesic paths might become more computationally efficient than using SVI.

References

Chen, N., Ferroni, F., Klushyn, A., Paraschos, A., Bayer, J., and Smagt, P. v. d. Fast approximate geodesics for deep generative models. In *International Conference on Artificial Neural Networks*, pp. 554–566. Springer, 2019.

Crane, K., Livesu, M., Puppo, E., and Qin, Y. A survey

of algorithms for geodesic paths and distances. *arXiv preprint arXiv:2007.10430*, 2020.

Das, A. and Rad, P. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*, 2020.

DeYoung, J., Jain, S., Rajani, N. F., Lehman, E., Xiong, C., Socher, R., and Wallace, B. C. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

Dombrowski, A.-K., Alber, M., Anders, C., Ackermann, M., Müller, K.-R., and Kessel, P. Explanations can be manipulated and geometry is to blame. *Advances in Neural Information Processing Systems*, 32, 2019.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Hassija, V., Chamola, V., Mahapatra, A., Singal, A., Goel, D., Huang, K., Scardapane, S., Spinelli, I., Mahmud, M., and Hussain, A. Interpreting black-box models: a review on explainable artificial intelligence. *Cognitive Computation*, 16(1):45–74, 2024.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jha, A., K Aicher, J., R Gazzara, M., Singh, D., and Barash, Y. Enhanced integrated gradients: improving interpretability of deep learning models using splicing codes as a case study. *Genome biology*, 21(1):1–22, 2020.

Kapishnikov, A., Venugopalan, S., Avci, B., Wedin, B., Terry, M., and Bolukbasi, T. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5050–5058, 2021.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Marcus, G. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Ribeiro, M. T., Singh, S., and Guestrin, C. " 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*, pp. 1135–1144, 2016.

Sap, M., Card, D., Gabriel, S., Choi, Y., and Smith, N. A. The risk of racial bias in hate speech detection. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 1668–1678, 2019.

Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In *International conference on machine learning*, pp. 3145–3153. PMLR, 2017.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Tenenbaum, J. B., Silva, V. d., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Tonekaboni, S., Joshi, S., Campbell, K., Duvenaud, D. K., and Goldenberg, A. What went wrong and when? instance-wise feature importance for time-series black-box models. *Advances in Neural Information Processing Systems*, 33:799–809, 2020.

Yang, T., Arvanitidis, G., Fu, D., Li, X., and Hauberg, S. Geodesic clustering in deep generative models. *arXiv preprint arXiv:1809.04747*, 2018.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

A. Additional Half-moons results

We present on Figure 8 more results on the half-moons dataset, using different amounts of noise. We can see that, for certain amount of noise, Enhanced IG dramatically fails, while Geodesic IG performs consistently well on every amount of noise tested here. We believe that the failure of Enhanced IG in the high-noise setting is due to the following reason. As noise increases the points on either moons get closer to each other. As a result, the model loses the property that gradients are only large on the decision boundary and fall rapidly as we move away. Therefore a lot of points are on the high-gradient regions. However, Enhanced IG chooses the path purely based on nearest neighbours, ignoring model gradients. Hence, leading to low purity. This is contrast to geodesic IG, which actively avoids regions of high gradients.

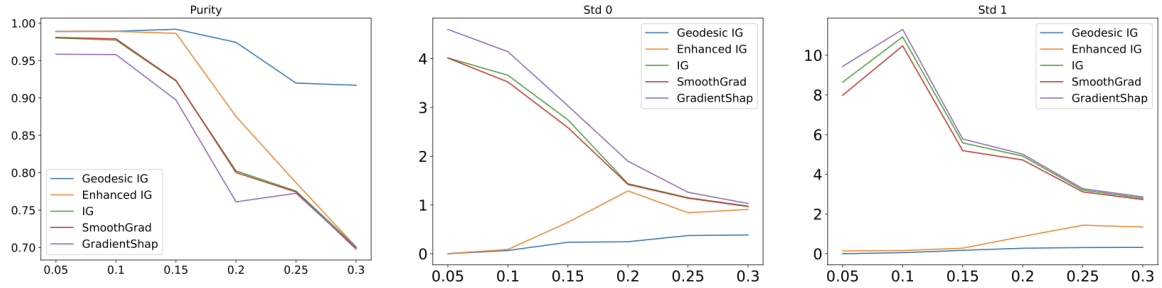

Figure 8. Evaluation of different attribution methods on the half-moons dataset with different amounts of noise: $\mathcal{N}(0, x)$ where x is defined as the axis of each plot.

We also perform here an ablation study using different values of k for the kNN algorithm. We show the results on Figure 9. The results show that increasing k harms the performance of Enhanced IG, leaving the ones of Geodesic IG unchanged. This is probably due to the fact that increasing k allows connections between points further apart, potentially crossing high-gradients regions. While Geodesic IG would not follow such paths, Enhanced IG only uses euclidean distance, and is therefore more likely to generate paths crossing high-gradients regions.

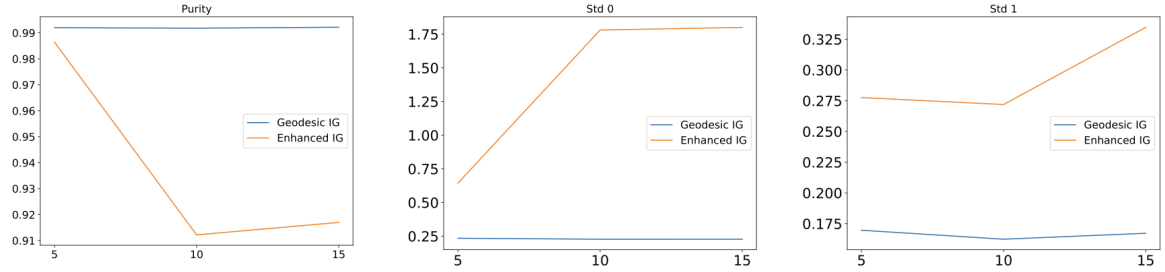

Figure 9. Evaluation of Geodesic IG and Enhanced IG for different values of k in the kNN algorithm. We can see that increasing this parameter harms Enhanced IG performance, while it does not seem to have a major effect on Geodesic IG performance.

B. Additional heatmaps and results on Pascal VOC 2012

We also qualitatively compare on Figure 10 Geodesic IG with the original IG on 5 different images of the Pascal VOC 2012 dataset. Geodesic IG heatmaps appears to be less blurry than the ones generated with original IG method.

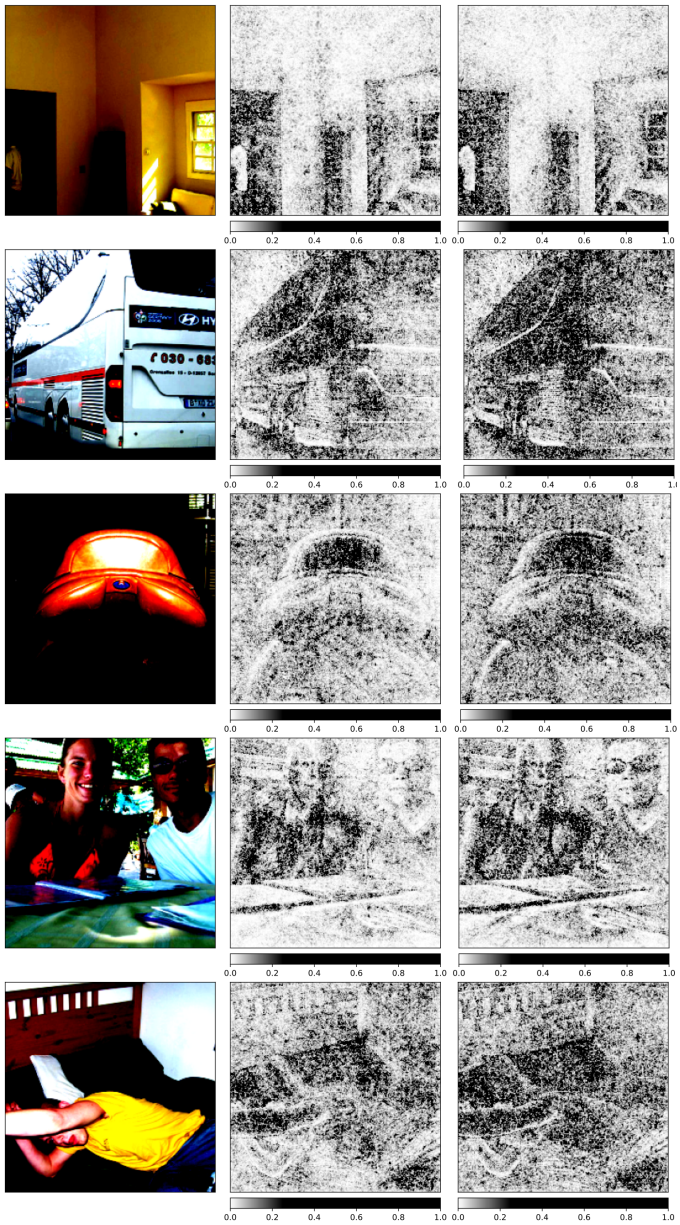

Figure 10. Heatmaps of Integrated Gradients (middle) and Geodesic IG (right) on 5 randomly chosen images from the test set of Pascal VOC 2012. We can see that Geodesic IG heatmaps are sharper compared with the original IG method.