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.

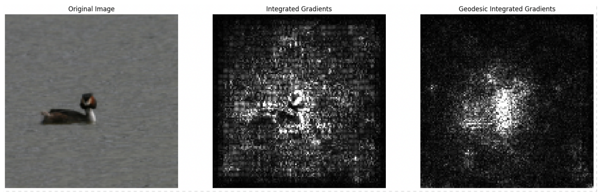

Figure 1. Comparison of attributions generated by Integrated Gradients (middle figure) and Geodesic Integrated Gradients (right figure) for image classification with a ConvNext model. Integrated Gradients follow straight paths in Euclidean space, which can result in misleading attributions. In contrast, Geodesic Integrated Gradients integrate along geodesic paths on a Riemannian manifold defined by the model, correcting the misattributions due to poor alignment with the model's gradient landscape.

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 $f : R^n \rightarrow R$, 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 approximating the following path integral

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

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 flawed explanations. We examine the consequences of these issues through examples in computer vision, as shown in Fig. 1, along with simpler, illustrative cases in Fig. 2. To address these challenges, we introduce **Geodesic Integrated Gradients** [1], a generalisation 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

[1]The code for the attribution methods and reproducing the experiments is provided on https://github.com/sina-salek/geodesic-ig

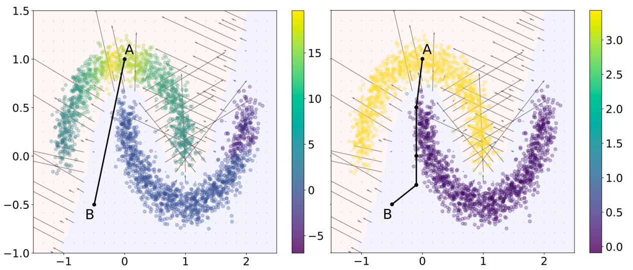

Figure 2. **Integrated Gradients (IG) attributions (left) vs. Geodesic IG (right).** The colour map shows feature attributions on the vertical axis, with $(-0.5, -0.5)$ (point B) as the baseline. In IG, points near A are over-attributed due to the straight path crossing high-gradient regions near the decision boundary (gray arrows). Geodesic IG avoids this issue by following geodesic paths, providing more accurate attributions.

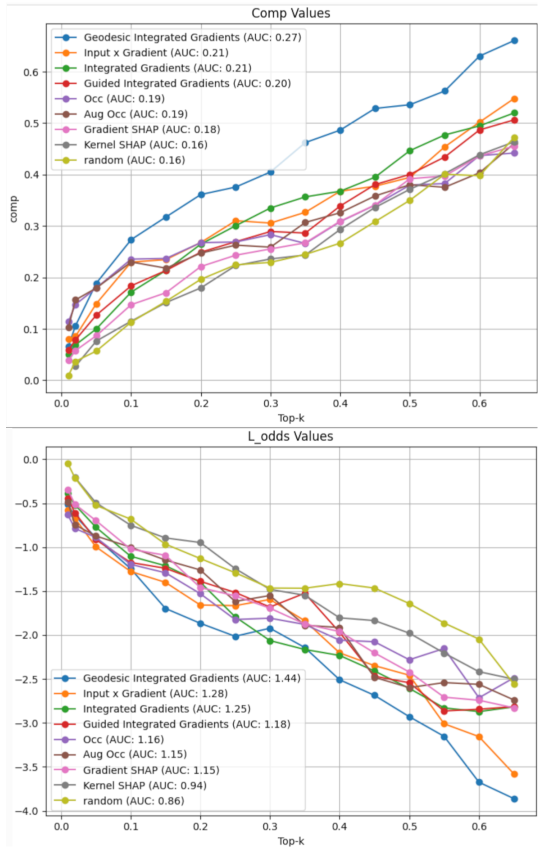

Figure 3. **Metrics Comparison.** We use a ConNext model for classification of images from the VOC dataset. The horizontal axis represents the top $k\%$ (in absolute value) of features being selected. The top plot, *Comprehensiveness*, shows the average change in the predicted class probability of compared to the original image (higher is better). The bottom plot displays the negative logarithmic probabilities of the predicted class relative to the original (lower is better). In both cases, the results are summarised using AUC, where higher values are better for both.

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

In Section 2, we present two methods for approximating the geodesic path between two points on a manifold. The first method, based on k-nearest neighbors (kNN), is designed for simpler manifolds, while the second method, utilising Stochastic Variational Inference, is suited for more complex manifolds. We further demonstrate that Geodesic IG adheres to all the axioms of Integrated Gradients.

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 two 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 $\mathbf{a}$ and $\mathbf{b}$ 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 $\mathbf{a}$ and $\mathbf{b}$ 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

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

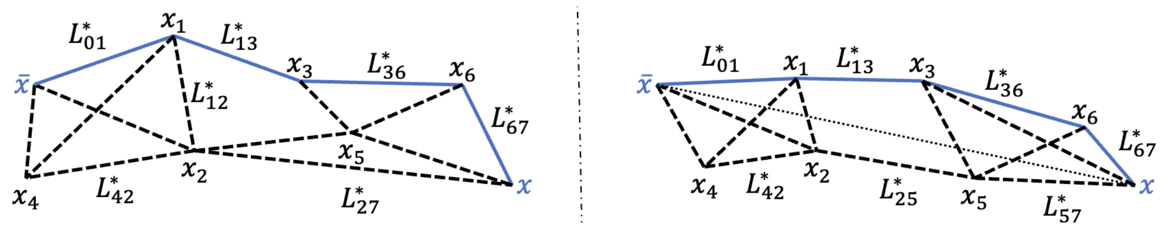

Figure 4. **Method overview.** For an input $\mathbf{x}$, a baseline $\bar{\mathbf{x}}$, and a set of points $\mathbf{x}_i$, we compute the kNN graph using the euclidean distance (dashed lines). For each couple $(\mathbf{x}_i, \mathbf{x}_j)$, we then compute the integrated gradients L_{ij}^* using Equation 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).

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

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

For input-baseline pair, $\mathbf{x}$ and $\bar{\mathbf{x}}$, we can now see the set $(\mathbf{x}, \bar{\mathbf{x}}, \mathbf{x}_i)$ 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:

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

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. See Fig. 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,

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

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.

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.

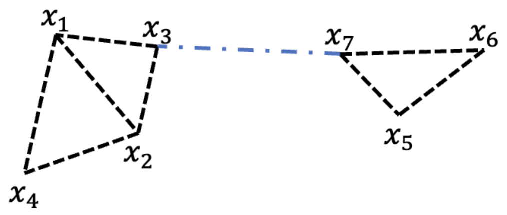

Figure 5. When the kNN graph is disconnected, as illustrated here, it would be impossible to compute Geodesic IG between certain points, for instance $\mathbf{x}_1$ and $\mathbf{x}_5$ here. To solve this, we add a single link between disconnected graphs, here between $\mathbf{x}_3$ and $\mathbf{x}_7$.

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, in this subsection, we devise an energy-base sampling method as another approximation procedure.

As noted in Remark 2.1, we aim to sample from the shortest paths between two points in the input space while avoiding regions of high gradients. To achieve this, we deviate from the straight line to minimise the influence of high-gradient areas. This process can be approximated as follows: we begin with a straight-line path between the two points and define a potential energy function composed of two terms—a distance term to maintain proximity to the straight line and a curvature penalty term to account for high gradients. Minimising this energy function approximates the geodesic path.

Formally, the distance term is defined as $d(\mathbf{x}, \mathbf{y}) := \|\mathbf{x} - \mathbf{y}\|_2$, and the curvature term as $c(\mathbf{x}) := \|\nabla f(\mathbf{x})\|_2$ where f represents the neural network. The total energy being minimised is

$$E(\gamma) = \sum_{i=1}^{n} d(\gamma_i, \gamma_i^0) - \beta c(\gamma_i), \tag{9}$$

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

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

SVI provides a probabilistic framework for optimising paths between input and baseline points. To achieve this, it 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. 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 apply this method in our computer vision experiments, demonstrating its efficacy in Section 3. However, for clarity, we visualise these paths on a simpler 2D half-moons example in Fig. 6. While the kNN method would typically be preferred for such simpler cases due to its ease of control, this example serves as an instructive illustration.

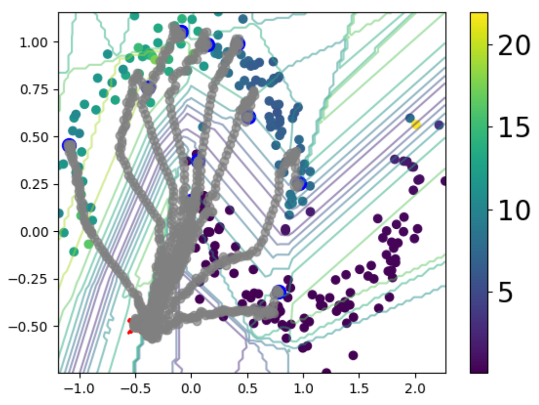

Figure 6. **Visualisation of 10 random paths.** For the simple case of half-moons with 1,000 samples, we display the sampled paths between 10 pairs of points. In low-gradient regions, the sampler favours straight lines, whereas in high-gradient regions, the paths adjust to become nearly perpendicular to the large gradient vectors, crossing these regions as quickly as possible.

Of course, using the SVI method comes with its own challanges. For example, in this case a suitable value of β for the potential energy, as well as learning rate for the SVI algorithm itself needs to be chose. As for any standard machine learinng training, these values can be chosen using hyperparameter tuning, as we discuss in section 3.

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 $\bar{\mathbf{x}}$, and a function f that is symmetric in dimensions i and j. If $\mathbf{x}_i = \mathbf{x}_j$ and $\bar{\mathbf{x}}_i = \bar{\mathbf{x}}_j$, 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-based attribution method that satisfies symmetry for any function. However, as noted in (Kapishnikov et al., 2021), while the straight path is the only one satisfying symmetry for *any* function, for a specific function, it may be possible to find other paths that also satisfy symmetry. Below, we demonstrate that Geodesic IG satisfies symmetry for Riemannian manifolds, and thus for the neural network functions we use when sampling the paths.

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 $\bar{\mathbf{x}}$ and U a Riemann neighbourhood centred at $\bar{\mathbf{x}}$. Let us also define the geodesic path γ such as $\gamma(0) = \bar{\mathbf{x}}$. 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

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 artefacts not reflecting the model's behaviour, and a low variation of attributions between close points. To that goal, we use **purity**, 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.

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. To provide better understand-

Method	AUC-Purity ↑
Input X Gradients	0.336
GradientShap	0.472
IG	0.476
Random	0.301
Kernel Shap	0.472
Occlusion	0.535
Guided IG	0.343
Geodesic IG (kNN)	**0.543**
Geodesic IG (SVI)	0.509

Table 1. Evaluation of different attribution methods on a half-moons dataset with Gaussian noise $\mathcal{N}(0, 0.2)$. We present in Appendix A more results with different amounts of Gaussian noise.

ing of these results we present more analysis on this dataset in Section 4.

3.2. Experiments on the Pascal VOC 2012 dataset

To evaluate our method on a real-world dataset, we used the Pascal VOC 2012 dataset (Everingham et al.), which consists of labeled images. We trained a classification head on this dataset and integrated it with the pre-trained ConvNext model (Liu et al., 2022) from TorchVision to generate predictions for explanation.

We applied the following attribution methods to explain the classifications for 100 randomly selected images: 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). For explainers that require a baseline, such as IG and our proposed method, a uniformly black image was used as the baseline.

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

- **Comprehensiveness** (DeYoung et al., 2019): We maks 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.

[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.

Method	AUC-Comp ↑	AOC-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
Guided IG	0.20	1.18
Geodesic IG (SVI)	**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 evaluated these metrics for a range of top-$k\%$, from 1% to 65%, as shown in Fig. 3. To summarise performance across different $k\%$ values, we calculated the area under the curve (AUC) for Comprehensiveness. We similarly calculated the area over the curve (AOC) for Log-odds, since the values of this metric are negative. In both cases these are the area between the metric curves and the horizontal line at $y = 0$. These results are presented in Table 2.

Additionally, Fig. 1 provides a qualitative comparison between Geodesic IG and Integrated Gradients. The results demonstrate that Geodesic IG outperforms other methods in explaining the model's behaviour on the dataset, with a particularly notable improvement in comprehensiveness. Further qualitative comparisons can be found in Appendix B.

Using Geodesic IG to explain complex deep learning models comes with several challenges. One major issue is the high computational cost of sampling methods like SVI or HMC. For instance, running the aforementioned experiment on 100 images required 23 hours on an L4 GPU. While computationally expensive, this could be justified in scenarios where accurate explanations are crucial. Another challenge is that the performance of the energy-based geodesic method heavily depends on selecting the right value for β in Eq. 9 and tuning SVI hyperparameters, such as the learning rate. In principle, hyperparameter tuning using metrics like Comprehensiveness or Log-odds could help optimise these parameters. However, due to limited computational resources, we were unable to perform such tuning in this study, though it has the potential to significantly improve results. Lastly, the optimisation nature of these sampling methods can cause the endpoints of the paths to deviate slightly from the baseline and input points, favoring nearby points in lower gradient regions. To address this, we we added a term to the potential energy to correct for these deviations and ensure more accurate alignment with the intended points. Formally, the extra term is $w \sum_{t \in \text{endpoints}} |\gamma(t) - \gamma_{\text{init}}(t)|_2$, where w is endpoint weight and the first/ last 10% of the paths are counted as endpoints .

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 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).

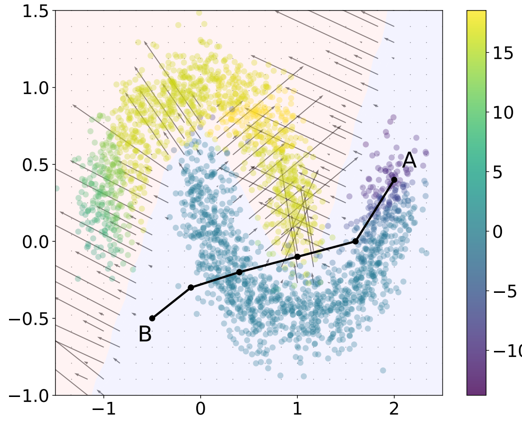

Figure 7. **Enhanced IG attributions.** Enhanced IG computes a kNN algorithm, uses Dijkstra to find the shortest path between an input and a reference baseline, and computes gradients along this path. However, this method is model agnostic and can as a result cross a high gradients region, which is the 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

In this paper, we identified potential issues with path-based attribution methods, particularly the artefacts arising from ignoring the curvature of the model. To address these challenges, we introduced Geodesic IG, an adaptation of the original IG method that integrates gradients along geodesic paths on a manifold defined by the model, rather than straight lines.

By avoiding regions of high gradient in the input space, Geodesic IG effectively mitigates these issues while adhering to all the axioms outlined by Sundararajan et al. (2017). We demonstrated its advantages through both theoretical analysis and empirical evaluation using metrics such as Comprehensiveness and Log-Odds.

To approximate geodesic paths, we proposed two methods: one based on kNN and the other on Stochastic Variational Inference (SVI). While these methods show clear advantages over alternatives, they also raise important questions for future research. One challenge is the computational cost, as discussed in Section 3. Another is the inherent noise in sampling-based approaches for approximating geodesic paths. A promising direction could involve solving the geodesic equation directly, which may reduce noise and provide more accurate path approximations. Additionally, depending on the chosen approach to solve these equations, it might be possible to achieve greater computational efficiency compared to the current reliance on 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.

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.

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.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

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.

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 blackbox 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.

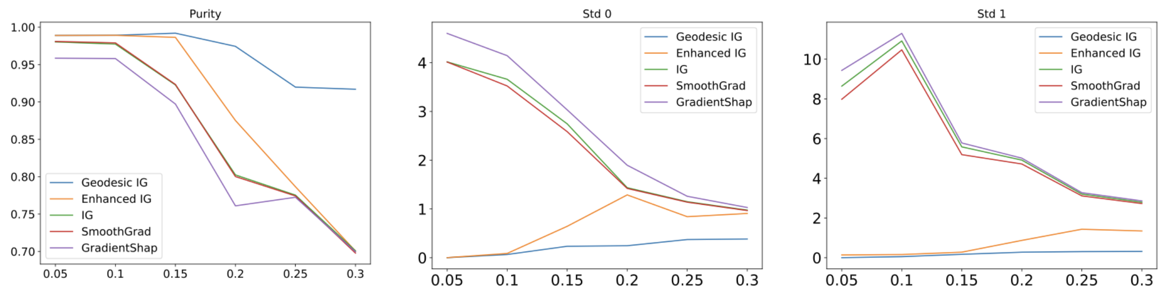

Figure 8. Evaluation of different attribution methods on the half-moons dataset with different amounts of noise: $\mathcal{N}(0, x)$ where x is defined as the axis of each plot.

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.

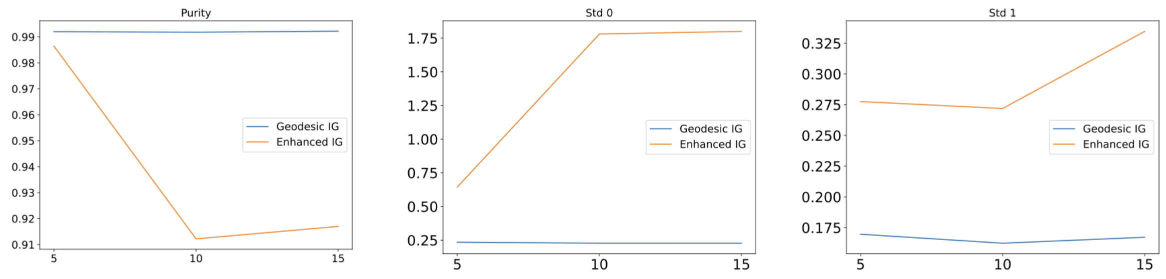

Figure 9. Evaluation of Geodesic IG and Enhanced IG for different values of k in the kNN algorithm. We can see that increasing this parameter harms Enhanced IG performance, while it does not seem to have a major effect on Geodesic IG performance.

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. In these images Geodesic IG heatmaps appears to to have fewer artefacts.

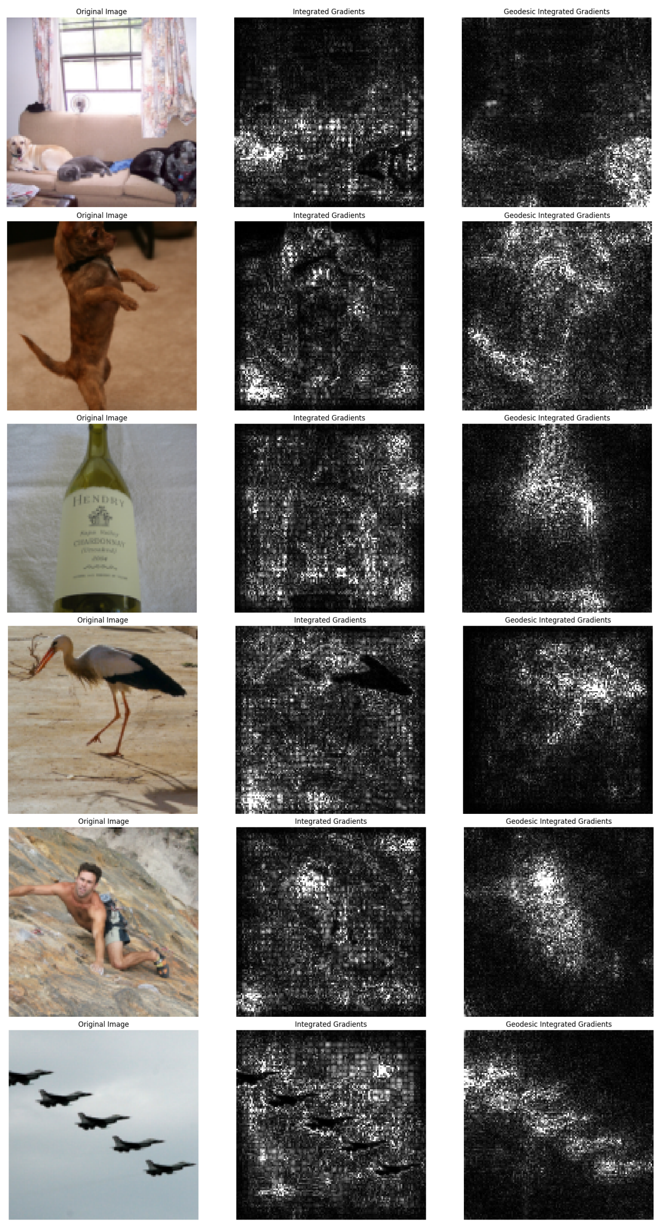

Figure 10. Heatmaps of Integrated Gradients (middle) and Geodesic IG (right) on 5 images from the test set of Pascal VOC 2012.