Negative Flux Aggregation to Estimate Feature Attributions

Xin Li , **Deng Pan** , **Chengyin Li** , **Yao Qiang** and **Dongxiao Zhu**

Department of Computer Science, Wayne State University, USA

{xinlee, pan.deng, cli, yao, dzhu}@wayne.edu

Abstract

There are increasing demands for understanding deep neural networks' (DNNs) behavior spurred by growing security and/or transparency concerns. Due to multi-layer nonlinearity of the deep neural network architectures, explaining DNN predictions still remains as an open problem, preventing us from gaining a deeper understanding of the mechanisms. To enhance the explainability of DNNs, we estimate the input feature's attributions to the prediction task using divergence and flux. Inspired by the divergence theorem in vector analysis, we develop a novel Negative Flux Aggregation (NeFLAG) formulation and an efficient approximation algorithm to estimate attribution map. Unlike the previous techniques, ours doesn't rely on fitting a surrogate model nor need any path integration of gradients. Both qualitative and quantitative experiments demonstrate a superior performance of NeFLAG in generating more faithful attribution maps than the competing methods. Our code is available at https://github.com/xinli0928/NeFLAG.

1 Introduction

The growing demand for trustworthy AI in security- and safety-critic domains has motivated developing new methods to explain DNN predicitons using image, text and tabular data [Chefer *et al.*, 2021; Li *et al.*, 2021; Pan *et al.*, 2020; Qiang *et al.*, 2022]. As noted in some pioneering works, e.g., [Hooker *et al.*, 2019; Smilkov *et al.*, 2017; Kapishnikov *et al.*, 2021], faithful explanation is indispensable for a DNN model to be trustworthy. However, it remains to be challenging for human to understand a DNN's predictions in terms of its input features due to its black-box nature.

As such, the field of explainable machine learning has emerged and seen a wide array of model explanation approaches. Among others, local approximation and gradient based methods are the two major categories that are more intensively researched. Local approximation methods mimic the local behavior of a black-box model within a certain neighborhood using some simple interpretable models, such as linear models and decision trees. However, they either require an additional model training processes (e.g., LIME [Ribeiro *et al.*,

2016], SHAP [Lundberg and Lee, 2017]), or rely on some customized propagation rules (e.g., LRP [Bach *et al.*, 2015], Deep Taylor Decomposition [Montavon *et al.*, 2017], DeepLIFT [Shrikumar *et al.*, 2017], DeepSHAP [Chen *et al.*, 2021]). Gradient based methods such as Saliency Map [Simonyan *et al.*, 2013], SmoothGrad [Smilkov *et al.*, 2017], FullGrad [Srinivas and Fleuret, 2019], Integrated Gradient (IG) and its variants [Sundararajan *et al.*, 2017; Hesse *et al.*, 2021; Erion *et al.*, 2021; Pan *et al.*, 2021; Kapishnikov *et al.*, 2019; Kapishnikov *et al.*, 2021] require neither surrogates nor customized rules but must tackle unstable estimates of gradients w.r.t. the given inputs. IG type of path integration based methods mitigate this issue via a path integration for gradient smoothing, however, this also introduces another degree of instability and noise sourced from arbitrary selections of baselines or integration paths. Others (e.g., Saliency Map, Full-Grad) relax this requirement nevertheless can be vulnerable to the small perturbations of the inputs due to its locality.

Ideally, we hope to avoid surrogates, special rules, arbitrary baselines and/or path integrals in interpreting DNN's prediction. In addition, since DNN interpretation often works on per sample basis, efficient algorithms are also crucial for a scalable DNN explanation technique. Here we examine the prediction behavior of DNNs through the lens of divergence and flux in vector analysis. We propose a novel Negative Flux Aggregation (NeFLAG) approach, which reformulates gradient accumulation as divergence. By converting divergence to gradient fluxes according to divergence theorem, NeFLAG interprets the DNN prediction with an attribution map obtained by efficient aggregation of the negative fluxes (see divergence theorem, flux, and divergence in Preliminaries).

We summarize our contributions as follows: 1) To the best of our knowledge, this is the first attempt to tackle the problem of explaining DNN model prediction leveraging the concepts of gradient divergence and fluxes. 2) Our NeFLAG technique eliminates the need for path integration via converting divergence to flux estimation, opening a new avenue for gradient smoothing techniques. 3) We propose an efficient approximation algorithm to enable a scalable DNN model explanation technique. 4) We investigate the relationship between flux estimation and Taylor approximation, bridging our method with other local approximation methods. 5) Both qualitative and quantitative experiment results demonstrate NeFLAG's superior performance to the competing methods in terms of

interpretation quality.

2 Related Work

We categorize our approach as one of the local approximation methods. Other examples include LIME [Ribeiro *et al.*, 2016], which fits a simple model (linear model or decision trees etc.) in the neighborhood of a given input point, and use this simple model as a local interpretation. Similarly, SHAP [Lundberg and Lee, 2017] generalizes LIME via a Shapley value reweighing scheme that has been proved to yield more consistent results. These methods, in general, enjoy their own merit when the underlying model is completely black-box, i.e., no gradient and model architecture information are available. However, in the case of DNN model interpretation, we usually know both model architecture and gradient information, so it is often beneficial to utilize this extra information as we are essentially interpreting the DNN model itself rather than a surrogate model. Saliency Map [Simonyan *et al.*, 2013] and FullGrad [Srinivas and Fleuret, 2019] exploit the gradient information, yet remain sensitive to small perturbations of the inputs. SmoothGrad [Smilkov *et al.*, 2017] improves it by averaging attributions over multiple perturbed samples of the input by adding Gaussian noise to the original input. As we will see in Section of Preliminaries below, our NeFLAG method not only exploits the smoothing gradient information to achieve robustness against perturbations, but also eliminates the need for fitting an extra surrogate model.

Another line of research explains DNN predictions via accumulating/integrating gradients along specific paths from baselines to the given input. Here we denote them as path integration based methods, represented by Integrated Gradients (IG) [Sundararajan *et al.*, 2017]. Recent variants includes fast IG [Hesse *et al.*, 2021], and IG with alternative paths and baselines (e.g., Adversarial Gradient Integration (AGI) [Pan *et al.*, 2021], Expected Gradient (EG) [Erion *et al.*, 2021]), Guided Integrated Gradients (GIG) [Kapishnikov *et al.*, 2021], Attribution through Regions (XRAI) [Kapishnikov *et al.*, 2019]). IG chooses a baseline (usually a black image) as the reference to calculate the attribution by accumulating gradients along a straight-line path from the baseline to the given input in the input space. AGI, on the other hand, relaxes the baseline and straight-line assumption by utilizing adversarial attacks to automatically generate both baselines and paths, thereafter accumulating gradients along these paths. Both would require paths and baselines for gradient smoothing via path integration regardless of manually picked or generated, which could introduce attribution noises along the path (as noted by [Sundararajan *et al.*, 2017], different paths could result in completely different attribution maps). On the contrary, our NeFLAG method doesn't need a path nor a baseline, in which the gradient smoothing is controlled by a single radius parameter ϵ, opening up a new direction for gradient smoothing techniques.

3 Preliminaries: Divergence and Flux

We start explaining our approach by introducing the concept of divergence and flux in vector analysis. Lets first consider a general scenario: to interpret a DNN's prediction, we hope to characterize its local behavior. Let's define a DNN model $f : \boldsymbol{X} \rightarrow \boldsymbol{Y}$, which takes inputs $\boldsymbol{x} \in \boldsymbol{X}$, and outputs $f(\boldsymbol{x}) \in \boldsymbol{Y}$. For simplicity, we also assume that the model is locally continuously differentiable. When we query the interpretation for a given input $\boldsymbol{x}$, we are interested in how and why a decision is made, i.e., what is the underlying decision boundary. In fact, this is also the idea behind many other interpretation methods such as LIME [Ribeiro *et al.*, 2016] and SHAP [Lundberg and Lee, 2017]. In these methods, an interpretable linear model (or other simple models) is fitted via sampling additional points around the neighborhood of input of interest. Clearly, taking advantage of a certain kind of neighborhood aggregation is a promising route for interpretation of local approximation.

When the gradients $\nabla_{\boldsymbol{x}} f$ are available, it is already a decent indicator of DNN's local behavior. If we only calculate the gradients at $\boldsymbol{x}$, the resulting attribution map is called the Saliency Map [Simonyan *et al.*, 2013]. However, without aggregation, these gradients are usually unstable due to adversarial effect, where a small perturbation of the inputs can lead to large variation of gradient values. On the other hand, these gradients may be vanishing due to the so-called saturation effect [Miglani *et al.*, 2020]. To overcome the instability, lets denote a neighborhood around input $\boldsymbol{x}$ to be $V_{\boldsymbol{x}}$, and estimate the average gradients over it. Intuitively, the resulting vector $\int_{V_{\boldsymbol{x}}} \nabla f \cdot \mathrm{d}V_{\boldsymbol{x}}$ can be viewed as a local approximation for the underlying model. However, neither a single gradient evaluation nor neighborhood gradient integration exploits the fact that the function value $f(\boldsymbol{x})$ can be viewed as a result of gradient field flow accumulation. The gradient field flow therefore inspires us to accumulate the gradients along the flow direction, giving rise to the new idea of gradient accumulation.

3.1 Gradient Accumulation

Plenty of methods have already embraced the idea of accumulation without explicitly defining it. They typically accumulate the gradients along a certain path, for example, Integrated Gradients (IG) [Sundararajan *et al.*, 2017] integrates the gradients along a straightline in the input space from a baseline to the given input. Specifically, they study not a single input point, but the gradient flows from a baseline point towards the given input (Figure 1). The baseline here is selected assuming no information is contained for decision. Another example is Adversarial Gradient Integration (AGI) [Pan *et al.*, 2021], which instead integrates along multiple paths generated by an adversarial attack algorithm, and aggregates all of them. This method also accumulates gradient flows, and it differs from IG only in accumulation paths and selection of baselines.

Although both IG and AGI exploit the idea of gradient accumulation, they only tackle one-dimensional accumulation, i.e., accumulating over a path/paths. In fact, [Pan *et al.*, 2021] point out that neither an accumulation path is unique, nor a single path is necessarily sufficient. Ideally, an accumulation method should take into account of every possible baseline and path. However, it is nearly impossible to accumulate all possible paths. Then how could we overcome such an obstacle? A promising direction is to examine the problem through the lens of divergence and flux.
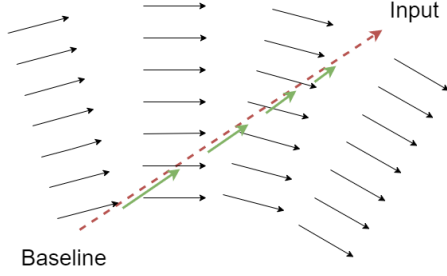
Figure 1: In a non-uniform field, IG accumulates gradient flow from a baseline to the given input. Here black solid arrows denote the gradient field, the red dashed arrow denotes the accumulation direction, and the length of green solid arrows denote the gradient magnitude on the accumulation direction.

3.2 Divergence and Flux

For studying accumulation, the definition of divergence perfectly fits our needs. Let $\mathbf{F} = \nabla_x f$ be the gradient field, the divergence div $\mathbf{F}$ is defined by

$$\text{div } \mathbf{F} = \nabla \cdot \mathbf{F} = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, ... \right) \cdot (F_{x_1}, F_{x_2}, ...), \quad (1)$$

where F_{x_i} and x_i are the gradient vector's and input's ith entry, respectively. The above definition is convenient for computation, however, difficult to interpret. A more intuitive definition of divergence can be described as follows.

Definition 1. *Given that $\mathbf{F}$ is a vector field, the divergence at position $\boldsymbol{x}$ is defined by the total vector flux through an infinitesimal surface enclosure S that encloses $\boldsymbol{x}$, i.e.,*

$$\text{div } \mathbf{F}|_{\mathbf{x}} = \lim_{V \to 0} \frac{1}{|V|} \oiint_{S(V)} \mathbf{F} \cdot \hat{\mathbf{n}} dS, \quad (2)$$

where V is an infinitesimal volume around $\boldsymbol{x}$ that is enclosed by S, and $\mathbf{F} \cdot \hat{\mathbf{n}}$ denotes the normal vector flow (flux) through the surface S.

It is not hard to see that $\nabla \cdot \mathbf{F}$ is essentially the (negatively) accumulated gradients at point $\boldsymbol{x}$ from Definition 1, as it defines the total gradient flow contained in the infinitesimal volume V.

Therefore, given input $\boldsymbol{x}$, to interpret the model within its neighborhood denoted by $V_{\boldsymbol{x}}$, we only need to calculate the accumulated gradients by integrating the divergence over all points within this neighborhood. The attribution heatmap can be obtained by simply replacing the dot product in Eq. 1 by an element-wise product, and then integrate, i.e.,

$$\text{Attribution} = \int_{V_{\boldsymbol{x}}} \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, ... \right) \odot (F_{x_1}, F_{x_2}, ...) \, dV. \tag{3}$$

Eqs. 2 and 3 are equivalent because of the divergence theorem described below. We use element-wise product because we want to disentangle the effects from different input entries.

However, there is one major obstacle, that is, how to integrate the whole volume $V_{\boldsymbol{x}}$ especially when the input dimension is high. It is tempting to sample a few points inside the neighborhood surface enclosure, and sum up the divergence.

But the computational cost for divergence, which involves second order gradient computation, is much higher than only calculating the first order gradients.

Fortunately, we can gracefully convert the volume integration of divergence into surface integration of gradient fluxes using divergence theorem, and thus simplify the computation.

3.3 Divergence Theorem

In vector analysis, the divergence theorem states that the total divergence within an enclosed surface enclosure is equal to the surface integral of the vector field's flux over such surface enclosure. The flux is defined as the vector field's normal component to the surface. Formally, let $\mathbf{F}$ be a vector field in a space U, S is defined as a surface enclosure in such space, and we call the neighborhood volume that is enclosed in S as V. Assuming that $\mathbf{F}$ is continuously differentiable on V, the *Divergence Theorem* states that

$$\iiint_V (\nabla \cdot \mathbf{F}) dV = \oiint_S (\mathbf{F} \cdot \hat{\mathbf{n}}) dS. \tag{4}$$

Here the integrating symbols $\iiint$ and $\oiint$ don't necessarily need to be 3-dimensional and 2-dimensional, respectively. They can be any higher dimension as long as the surface integration is one dimension lower than the volume integration.

In DNNs, the gradient field $\mathbf{F} = \nabla_{\boldsymbol{x}} f$ w.r.t. the inputs is exactly a vector field in the input space. Note that there could be non-differentiable points (that could violate the continuously differentiable condition due to some activation functions such as ReLU may not differentiable at some point). Nevertheless we can safely assume that it is at least continuously differentiable within a certain small neighborhood.

4 Negative Flux Aggregation

In this Section, we describe our interpretation approach, Negative Flux Aggregation (NeFLAG). We first explain the necessity of differentiating positive and negative fluxes, and then describe an algorithm on how to find negative fluxes on a ϵ-sphere and aggregate them for interpretation.

4.1 Negative Fluxes Define a Linear Model

Interpreting via directly integrating all fluxes is an intriguing approach, however, the positive and negative fluxes should be interpreted differently. By convention, positive fluxes are pointed outward the enclosure surface, and the negative fluxes are pointed inward (Figure 2a). Let S be a surface enclosure and V is its corresponding volume, a positive flux can be viewed as the gradient loss of V, and similarly a negative flux is the gradient gain of V. It means the confidence gain or loss of the prediction of $\boldsymbol{x}$. For example, from $f(\boldsymbol{x}) = 0.99$ to $f(\boldsymbol{x}) = 0.8$ represents a confidence loss. The sum of negative fluxes here means a total confidence loss if $\boldsymbol{x}$ moves out of the neighborhood. When interpreting a DNN model prediction f, assuming $\boldsymbol{x}$ is a point in the input space U, the prediction of it by f is class A. To interpret this prediction at a location in the input space (i.e., $\boldsymbol{x}$), we need to find out that moving to which direction could make it less likely to be A? Apparently, a good choice is to move in the direction of negative gradient, i.e., with negative gradient fluxes. In contrast, the positive flux

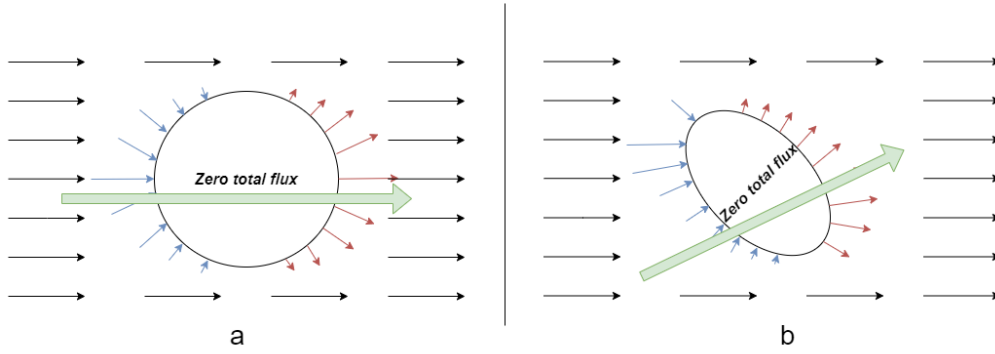a   b

Figure 2: The fluxes on closed surfaces with different shapes. **a**. The total flux of a closed sphere surface is zero when the field is uniform, and the vector obtained by negative flux vector aggregation is coherent to the field vector. Here red arrows and blue arrows denote positive and negative flux. The green arrow bar denotes the direction of the aggregated vector. **b**. When the closed surface is not symmetric with respect to the field, the resulting aggregated vector is not in the same direction as the field.

can be interpreted as what makes it more likely to be class A. But since x is already predicted as A, this information is less informative than the negative fluxes in terms of interpretation. (An additional intuition is that for a given input x we are more interested in its direction towards decision boundary, instead of the other side of it.)

Linear case. It is not hard to see that in a linear vector field $\mathbf{F}_l$ with underlying linear function f_l, the sum of fluxes within any enclosure is 0 (Figure 2). Moreover, when the enclosure is an N-dimensional sphere (where N is the dimension of the space), we can sum up over all vectors on the sphere surface that are contributing negative flux. The obtained summed-up vector is then equivalent to the weight vector of the underlying linear function f_l.

The observation of the linear case study inspires us to learn the weight vector of a local linear function by summing up all negative fluxes on an N-d sphere. Hence we say that the vector sum of all negative fluxes interprets the local behavior of the original function via a local linear approximation. Formally, we state Assumption 1 as follows.

Assumption 1. *Given model f whose gradient field is $\mathbf{F}$, and sample x, let $S_{\boldsymbol{x}}$ be an N-d ϵ-sphere that centered at $\boldsymbol{x}$. The vector sum of all negative flux can be obtained by*

$$\boldsymbol{w} = \oiint_{S_{\boldsymbol{x}}^-} \mathbf{F} \odot \hat{\mathbf{n}} dS_{\boldsymbol{x}}, \tag{5}$$

where $\odot$ denotes element-wise product, and $S_{\boldsymbol{x}}^-$ is a set of point on the sphere where flux is negative. The linear model defined by weight vector $\boldsymbol{w}$ is then a local linear interpretation to the original model of input $\boldsymbol{x}$.

Note that this assumption is derived from the observation when $y = a_1 x_1 + a_2 x_2 + ... + a_n x_n$, so we interpret this linear model by attributions $\mathbf{attr} = (a_1 x_1, a_2 x_2, ..., a_n x_n)$. The entry $attr_i = a_i x_i$ represents the contribution of the ith attribute to the final prediction. Note that the attribution can also be written as $\mathbf{attr} = \boldsymbol{a} \odot \boldsymbol{x}$, hence the element-wise product in the Assumption. It is also critical to set $S_{\boldsymbol{x}}$ as a sphere in Assumption 1, any other shape introducing any degree of asymmetry would cause disagreement between local

linear approximation and the negative flux aggregation, as shown in Figure 2b.

Interpretation. An interpretation can be achieved by plotting an attribution map using the vector $\boldsymbol{w}$ from Eq. 5. The rationale is directly from the correspondence between this vector and the underlying linear model.

4.2 An Approximation Algorithm

To calculate Eq. 5, we have to find out both the point $\tilde{\boldsymbol{x}}$ with negative flux and its normal vector $\hat{\mathbf{n}}$. For the latter, given a candidate point $\tilde{\boldsymbol{x}}$ on the sphere surface $S_{\boldsymbol{x}}$, we can replace $\hat{\mathbf{n}}$ by $(\tilde{\boldsymbol{x}} - \boldsymbol{x})/|\tilde{\boldsymbol{x}} - \boldsymbol{x}|$ because $\boldsymbol{x}$ is the center of the sphere $S_{\boldsymbol{x}}$, and the radius $\epsilon = |\tilde{\boldsymbol{x}} - \boldsymbol{x}|$. As for point $\tilde{\boldsymbol{x}}$ with negative flux, a straightforward solution is to randomly subsample a list of points on the ϵ-sphere, then select those with negative flux. However, this trick has no guarantee on how many subsampling is sufficient, and also cannot guarantee that a negative flux exists. Therefore, we need an approximation algorithm that can guide us to those points with negative fluxes.

Since we are interested in interpreting the local behavior, meaning that the radius of ϵ-sphere should be sufficiently small. This provides us with the possibility to approximate the gradient (flux). i.e.,

$$\mathbf{F}(\tilde{\boldsymbol{x}}) \cdot \hat{\mathbf{n}} \approx \frac{(f(\tilde{\boldsymbol{x}}) - f(\boldsymbol{x}))}{\epsilon}. \tag{6}$$

Since $\boldsymbol{x}$ is the center of ϵ-sphere $S_{\boldsymbol{x}}$, to make the flux (left hand side of Eq. 6) negative, we only need to find $\tilde{\boldsymbol{x}}$ such that $f(\tilde{\boldsymbol{x}}) < f(\boldsymbol{x})$. Moreover, for interpreting a classification task, it is usually the case that $f(\boldsymbol{x})$ is of a high value (when a prediction is confident, the output probability would be close to 1). Therefore, if we could find a random local minimum $\tilde{\boldsymbol{x}}$ on the ϵ-sphere $S_{\boldsymbol{x}}$, it would most likely have $f(\tilde{\boldsymbol{x}}) < f(\boldsymbol{x})$.

In order to obtain a local minimum starting with any arbitrary initialization, we propose the following Lemma.

Lemma 1. *Let $\boldsymbol{x}$ be the center of sphere $S_{\boldsymbol{x}}$, and $\boldsymbol{x}^{(0)}$ be a random point on the ϵ-sphere $S_{\boldsymbol{x}}$. We define the following recurrence*

$$\boldsymbol{x}^{(k)} = \boldsymbol{x} - \epsilon \cdot \frac{\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)}{\left|\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)\right|}. \tag{7}$$

Let V_x be the space enclosed by $S_{\boldsymbol{x}}$, assuming $\mathbf{F}$ is continuous on V_x, then $f(\boldsymbol{x}^{(k)})$ converges to a local minima on $S_{\boldsymbol{x}}$ as $k \to \infty$.

Proof. To find a local minima on the surface $S_{\boldsymbol{x}}$, given the current candidate $\boldsymbol{x}^{(k-1)}$ and its corresponding gradient $\mathbf{F}(\boldsymbol{x}^{(k-1)})$, the updating formula needs to take $\boldsymbol{x}^{(k-1)}$ to $\boldsymbol{x}^{(k)}$ such that $\boldsymbol{x}^{(k)}$ is on the opposite direction of the tangent component of $\mathbf{F}(\boldsymbol{x}^{(k-1)})$ to the surface $S_{\boldsymbol{x}}$. Assuming the updating step to be ϵ, the updating rule is then

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \epsilon \cdot \left(\frac{\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)}{\left|\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)\right|} - \frac{\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right) \cdot \hat{\mathbf{n}}}{\left|\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)\right|} \cdot \hat{\mathbf{n}}. \right). \tag{8}$$

Note that the term in the parentheses is exactly the tangent direction of the gradient (i.e., the direction of the gradient minus the normal component). We can derive that

$$\boldsymbol{x} = \boldsymbol{x}^{(k-1)} + \epsilon \cdot \frac{\left(\boldsymbol{x} - \boldsymbol{x}^{(k-1)}\right)}{\epsilon} \tag{9}$$

$$\approx \boldsymbol{x}^{(k-1)} + \epsilon \cdot \frac{\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right) \cdot \hat{\mathbf{n}}}{\left|\mathbf{F}\left(\boldsymbol{x}^{(k-1)}\right)\right|} \cdot \hat{\mathbf{n}}., \tag{10}$$

where $\boldsymbol{x}$ is the center of the ϵ-sphere $S_{\boldsymbol{x}}$, substituting Eq. 10 into Eq 8, we then have the updating rule in Eq. 7. $\square$

Lemma 1 tells us that given any initial point $\boldsymbol{x}^{(0)}$ on the sphere, we can always find a point with local minimal flux value. Ideally, by seeding multiple initial points, following the recurrence in Eq. 7, we can obtain a set of local minima points on $S_{\boldsymbol{x}}$. The integration in Eq. 5 can then be approximated by summation of the gradient fluxes at these local minima points. But this approximation is suboptimal because the updating rule in Eq. 8 will cause bias towards those points with *minimal fluxes*, instead of distributing evenly to points with *negative fluxes*. To overcome this issue, we add a sign operation on the recurrence in Eq. 7, implemented in Algorithm 1, to introduce additional stochasticity, hence making the negative flux points distributed more evenly. Algorithm 1 describes this trick as well as the step-by-step procedure of calculating NeFLAG.

Here we note Eq. 5 provides a *formulation* (or a *framework*) to calculate a representing vector for the neighborhood near $\boldsymbol{x}$. We develop Algorithm 1 as *one way of approximation* to efficiently perform the experiments. There are other algorithms with better approximation accuracy that we may explore the different approximation tricks in the future.

4.3 Connection to Taylor Approximation

NeFLAG can be viewed as a generalized version of Taylor approximation. Here we show that NeFLAG can not only be viewed as a gradient accumulation method, but also an extension to some local approximation methods. Given that $S_{\boldsymbol{x}}$ is an N-d ϵ-sphere centered at $\boldsymbol{x}$, for any point $\tilde{\boldsymbol{x}}$ on the sphere surface, we can represent the normal vector $\hat{\mathbf{n}}$ as $(\tilde{\boldsymbol{x}} - \boldsymbol{x})/\epsilon$. This inspires us to investigate its connection to the first order Taylor approximation. As pointed out by [Montavon *et al.*,

---

Algorithm 1 NeFLAG($f, \boldsymbol{x}, n, S_{\boldsymbol{x}}, \epsilon, m$)

Input: f: Classifier, $\boldsymbol{x}$: input, n: number of negative flux samples, $S_{\boldsymbol{x}}$: ϵ-sphere, ϵ: radius of $S_{\boldsymbol{x}}$, m: max number of backpropagation steps;
Output: Attribution map NeFLAG;
1: NeFLAG $\leftarrow 0$; $j \leftarrow 0$; $i \leftarrow 0$;
2: **while** $i = 1 : n$ **do**
3:   **while** $j = 1 : m$ **do**
4:     Randomly sample $\tilde{\boldsymbol{x}}$ on sphere $S_{\boldsymbol{x}}$;
5:     $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \epsilon \cdot \text{sign}\left(\frac{\mathbf{F}(\tilde{\boldsymbol{x}})}{|\mathbf{F}(\tilde{\boldsymbol{x}})|}\right)$;
6:   **end while**
7:   NeFLAG = NeFLAG + $\mathbf{F}(\tilde{\boldsymbol{x}}) \odot (\boldsymbol{x} - \tilde{\boldsymbol{x}})$;
8: **end while**

2017], Taylor decomposition can be applied at a nearby point $\tilde{x}$, then the first order decomposition can be written by

$$f(\boldsymbol{x}) = f(\tilde{\boldsymbol{x}}) + (\nabla_{\boldsymbol{x}} f|_{\boldsymbol{x}=\tilde{\boldsymbol{x}}})^{\top} \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}}) + \eta, \tag{11}$$

where η is the error term of second order and higher. We use $R(\boldsymbol{x}) = (\nabla_{\boldsymbol{x}} f|_{\boldsymbol{x}=\tilde{\boldsymbol{x}}}) \odot (\boldsymbol{x} - \tilde{\boldsymbol{x}})$ to represent a heatmap generated by this approximation process. Note that the heatmap $R(\boldsymbol{x})$ completely attributes $f(\boldsymbol{x})$ if $f(\tilde{x}) = 0$ and the error term η can be omitted.

One may notice that if $\tilde{x}$ is by chance on the N-d ϵ-sphere, the first order Taylor approxmation of $\boldsymbol{x}$ from $\tilde{x}$ is then equivalent to the flux estimation at $\tilde{x}$. To explain it more clearly, we can simply view $-(\boldsymbol{x} - \tilde{\boldsymbol{x}})/|\boldsymbol{x} - \tilde{\boldsymbol{x}}|$ as the normal vector $\hat{\mathbf{n}}$ in Eq. 5, and $(\nabla_{\boldsymbol{x}} f|_{\boldsymbol{x}=\tilde{\boldsymbol{x}}})$ is exactly the gradient vector $\mathbf{F}$ at $\tilde{x}$. Hence the first order Taylor decomposition and a single point flux estimation differs only by a factor of $-\epsilon$ as $\epsilon = |\boldsymbol{x} - \tilde{\boldsymbol{x}}|$.

Since we only care about the negative fluxes, then the second term of Eq 11, i.e., $(\nabla_{\boldsymbol{x}} f|_{\boldsymbol{x}=\tilde{\boldsymbol{x}}})^{\top} \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}})$ must be positive. If we further omit the error term η, we must have $f(\boldsymbol{x}) = f(\tilde{x}) + $ positive value. It tells us that in the perspective of first order Taylor decomposition, the negative flux indeed attributes to positive predictions, bridging the connection between NeFLAG and other local approximation methods.

5 Experiments

In this Section, we demonstrate and evaluate NeFLAG's performance both qualitatively and quantitatively using a diverse set of DNN architectures and baseline interpretation methods.

5.1 Experiment Setup

Models. InceptionV3 [Szegedy *et al.*, 2015], ResNet152 [He *et al.*, 2015] and VGG19 [Simonyan and Zisserman, 2014] are selected as the pre-trained DNN models to be explained.

Dataset. The ImageNet [Deng *et al.*, 2009] dataset is used for all of our experiments. ImageNet is a large and complex data set (compared with smaller and simpler data sets such as Places, CUB200, or Flowers102) for us to better demonstrate the key advantages of our approach.

Baseline interpretation methods. IG [Sundararajan *et al.*, 2017] and AGI [Pan *et al.*, 2021] are selected as baselines for both qualitative and quantitative interpretation performance
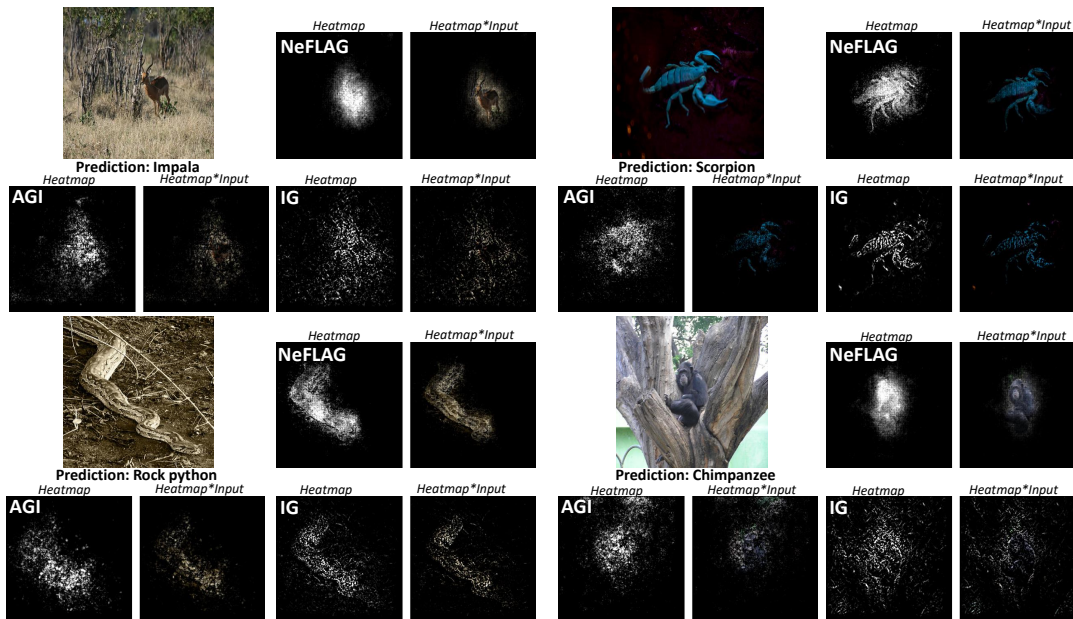
Figure 3: Examples of attribution maps obtained by NeFLAG, AGI and IG methods. The underlying prediction model is InceptionV3 (additional examples for ResNet152 and VGG19 can be found in the supplementary materials). Compared to AGI and IG, we observe that NeFLAG's attribution map has a clearer shape and focuses more densely on the target object.

Metrics	Methods/Models	IG	SG	GIG	EG	AGI	NeFLAG-1	NeFLAG-10	NeFLAG-20
Deletion Score ↓	VGG19	0.071	0.065	0.054	0.041	0.040	**0.034**	0.052	0.059
	ResNet152	0.132	0.091	0.090	0.066	0.068	**0.055**	0.076	0.081
	InceptionV3	0.122	0.079	0.096	0.049	0.059	**0.048**	0.066	0.068
Insertion Score ↑	VGG19	0.223	0.312	0.304	0.338	0.401	0.416	0.521	**0.535**
	ResNet152	0.332	0.380	0.437	0.447	0.480	0.485	0.568	**0.578**
	InceptionV3	0.375	0.465	0.465	0.478	0.480	0.544	0.618	**0.625**

Table 1: Quantitative evaluation using deletion and insertion scores. NeFLAG-1, NeFLAG-10 and NeFLAG-20 are NeFLAG methods with various number of negative flux samples. Note **NeFLAG-1** already **outperforms all** the competing methods with a **linear computational complexity** of the number of backprogation steps (see Appendix on Computational Complexity and Overhead for detailed analysis).

comparison. We also include Expected Gradients (EG)[Erion *et al.*, 2021], Guided Integrated Gradient [Kapishnikov *et al.*, 2021], and SmoothGrad [Smilkov *et al.*, 2017] in quantitative comparison as they can be viewed as smoothed versions of attribution methods. We use the default setting provided by captum [1] for IG, EG and SmoothGrad methods. For AGI, we adopt the default parameter settings reported in [Pan *et al.*, 2021], i.e., step size $\epsilon = 0.05$, number of false classes $n = 20$. For GIG, we also used the default settings provided in their latest official implementation [Kapishnikov *et al.*, 2021]. Here we focus on the comparison with *gradient based methods* since non-gradient methods typically require a surrogate and additional optimization processes.

5.2 Qualitative Evaluation

We first show some examples to demonstrate a better quality of the attribution maps generated by NeFLAG than by other

_____
[1] https://captum.ai/

baselines. The NeFLAG is configured as follows: ϵ-sphere radius is set to $\epsilon = 0.1$, and the number of random negative flux point $\tilde{x}$ is $n = 20$. Figure 3 shows the attribution maps generated by various interpretation methods for the Inception V3 model (examples for ResNet152 and VGG19 models can be found in the Appendix). Qualitatively speaking, we denote an attribution map is faithful to model's prediction if 1) it focuses on the target objects, and 2) has clear and defined shapes w.r.t. class label instead of sparsely distributed pixels. Based on this notion, it is clearly observed that NeFLAG has a better attribution heatmap quality than AGI and IG.

A key observation from Figure 3 is that the NeFLAG attribution heatmap provides a defined shape of the entity of the class whereas IG and AGI attribution heatmaps do not reflect that ability. In fact, the latter heatmaps are scattered and do not retain a definite shape. For example, in Figure 3, IG and AGI's heatmaps fail to delineate a definite and slender shape of the *Rock Python*. The same notion happens with the *Impala* and *Chimpanzee* where the attribution heatmaps of IG
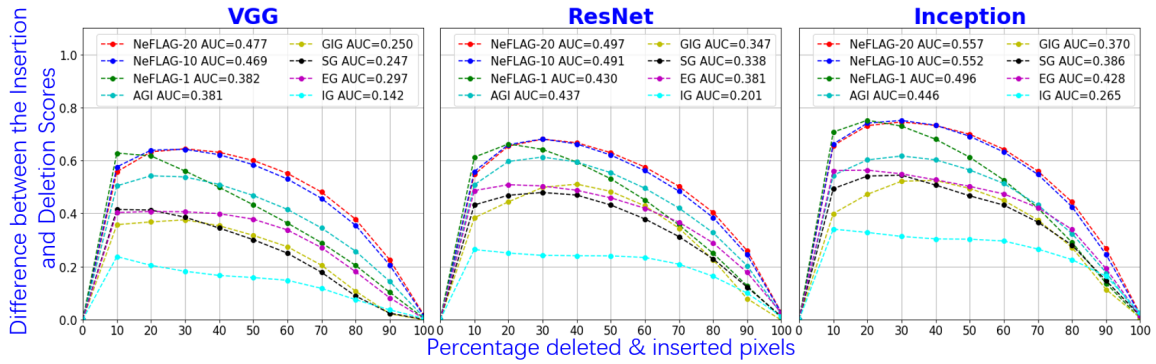
Figure 4: Quantitative Performance Comparison in terms of Difference between the Insertion and Deletion Scores.

and AGI are scattered across the background and class entity. We note that the results shown in Figure 3 are very typical (not cheery-picked) in our experiments. We also provide more examples from each class to demonstrate the superior quality of attribution maps in the Appendix.

We note that the striking performance disparities referenced above could mainly be caused by the inconsistency of path accumulation methods. Since both IG and AGI incorporate a certain kind of path integration, any gradient vector on that path is taken into account of the final interpretation. However, the gradient vectors on the path aren't necessarily unique and consistent. The rationale behind this argument is that neither IG nor AGI is capable of guaranteeing that their choice of accumulation path is optimal. For example, in IG, the path is a straight-line, which isn't by any means ideal. Similarly in AGI, the path of an adversarial attack is chosen. However, due to the stochastic effect, a slightly different initialization could result in completely different attack paths. Our method NeFLAG, on the other hand, takes advantage of negative flux aggregation without the burden of path accumulation.

In fact, NeFLAG's advantage is highly pronounced when we set the number of negative flux points to 1 (i.e., $n = 1$), where we will show hereinafter using quantitative experiments, is sufficient for a superior performance to the competing methods. Yet, NeFLAG-1 achieves an unequaled computational efficiency via a single point gradient evaluation instead of a path integration used in IG type methods.

5.3 Quantitative Evaluation

In the quantitative experiments, we compare the performance of IG, AGI, EG, GIG, SmoothGrad and three NeFLAGs variants with different numbers of sampled negative flux points. Since our NeFLAG is considered as a *robust local attribution* method, the size the neighborhood (radius of ϵ-sphere) and number of negative fluxes are tuning parameters. Similarly, the baseline local attribution method, SmoothGrad, also has two tuning parameters: the noise level and the number of samples to average over. Contrastively, IG, AGI, EG, and GIG are *global attribution* methods so there is no tuning parameter involved. We extensively investigate the neighborhood size and experiment with various number of negative fluxes (e.g., NeGLAG-1, NeGLAG-10, and NeGLAG-20) to faithfully demonstrate the stability of our method in comparison

with others.

Here VGG19, InceptionV3, and ResNet152 are used as underlying DNN prediction models. Since we focus on explaining DNN prediction on a *per sample basis* just like other DNN explanation methods, we randomly select 5,000 samples from ImageNet validation dataset with 5 samples from each class as a good representation of the classes. We use insertion score and deletion score as our evaluation metrics [Petsiuk *et al.*, 2018]. We replace the top pixels with black pixels in the first round [Petsiuk *et al.*, 2018] whereas with Gaussian blurred pixels in the second round [Sturmfels *et al.*, 2020]. We report the average performance of two rounds of experiments. Experimental details can be found in the Appendix.

Table 1 demonstrates that NeFLAG outperforms other baselines even with only a single negative flux point (NeFLAG-1), and becomes much better when we incorporate sufficient amount of negative flux points (NeFLAG-10 and NeFLAG-20). In Figure 4, we systematically compared the performance of our NeFLAG with the competing methods on three DNN architectures. We used the Difference between the Insertion and Deletion scores as the metric for comparison [Shah *et al.*, 2021]. It is because the deletion and insertion scores can be influenced by distribution shift caused by removing/adding pixels [Hooker *et al.*, 2019]. Since the shift occurs during both pixel insertion and deletion, focusing on their relative difference instead of their absolute values helps in neutralizing the effects of distribution shift. It is clearly observed from Figure 4 that our NeFLAG method outperforms all the competing methods across the three DNN models in terms of the Difference between the Insertion and Deletion scores. We note there still a room for more reliable and comprehensive evaluation metrics for the attribution methods.

6 Conclusion

We develop a novel DNN model explanation technique NeFLAG built off the concept of divergence, and point out a connection to other local approximation methods. NeFLAG doesn't require a baseline, nor an integration path. This is achieved by converting a volume integration of the second order gradients to a surface integration of the first order gradients using divergence theorem. Both qualitative and quantitative experiments demonstrate a superior performance of NeFLAG in explaining DNN predictions over the strong baselines.

References

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

[Chefer *et al.*, 2021] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 782–791, 2021.

[Chen *et al.*, 2021] Hugh Chen, Scott M Lundberg, and Su-In Lee. Explaining a series of models by propagating shapley values. *arXiv preprint arXiv:2105.00108*, 2021.

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

[Erion *et al.*, 2021] Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature Machine Intelligence*, pages 1–12, 2021.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.

[Hesse *et al.*, 2021] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. Fast axiomatic attribution for neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[Hooker *et al.*, 2019] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32, 2019.

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

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

[Li *et al.*, 2021] Xin Li, Xiangrui Li, Deng Pan, and Dongxiao Zhu. Improving adversarial robustness via probabilistically compact loss with logit constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8482–8490, 2021.

[Lundberg and Lee, 2017] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[Miglani *et al.*, 2020] Vivek Miglani, Narine Kokhlikyan, Bilal Alsallakh, Miguel Martin, and Orion Reblitz-Richardson. Investigating saturation effects in integrated gradients. *arXiv preprint arXiv:2010.12697*, 2020.

[Montavon *et al.*, 2017] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[Pan *et al.*, 2020] Deng Pan, Xiangrui Li, Xin Li, and Dongxiao Zhu. Explainable recommendation via interpretable feature mapping and evaluation of explainability. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2690–2696. ijcai.org, 2020.

[Pan *et al.*, 2021] Deng Pan, Xin Li, and Dongxiao Zhu. Explaining deep neural network models with adversarial gradient integration. In *Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

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

[Qiang *et al.*, 2022] Yao Qiang, Deng Pan, Chengyin Li, Xin Li, Rhongho Jang, and Dongxiao Zhu. Attcat: Explaining transformers via attentive class activation tokens. In *Advances in Neural Information Processing Systems*, 2022.

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

[Shah *et al.*, 2021] Harshay Shah, Prateek Jain, and Praneeth Netrapalli. Do input gradients highlight discriminative features? *Advances in Neural Information Processing Systems*, 34:2046–2059, 2021.

[Shrikumar *et al.*, 2017] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2017.

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

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

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

[Srinivas and Fleuret, 2019] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network

visualization. *Advances in neural information processing systems*, 32, 2019.

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

[Sundararajan *et al.*, 2017] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[Szegedy *et al.*, 2015] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. corr abs/1512.00567 (2015), 2015.