Pattern-Guided Integrated Gradients

Robert Schwarzenberg*[1] Steffen Castle*[1]

Abstract

Integrated Gradients (IG) and PatternAttribution (PA) are two established explainability methods for neural networks. Both methods are theoretically well-founded. However, they were designed to overcome different challenges. In this work, we combine the two methods into a new method, *Pattern-Guided Integrated Gradients* (PGIG). PGIG inherits important properties from both parent methods and passes stress tests that the originals fail. In addition, we benchmark PGIG against nine alternative explainability approaches (including its parent methods) in a large-scale image degradation experiment and find that it outperforms all of them.

1. Introduction

Integrated Gradients (Sundararajan et al., 2017) is a gradient-based explainability method with a sound theoretical motivation that has also performed well experimentally (Ancona et al., 2018). The authors of the PatternAttribution method (Kindermans et al., 2018), however, compellingly argue that IG does not give enough importance to the class signals in the input data (as do several other gradient-based explainability methods). They show that model weights function to cancel the distractor (noise) in the data and thus are more informative about the distractor than the signal. The weights even tend to direct the gradient (and hence attributions of gradient-based explainability methods) away from the signal. To direct the weights (and thus attributions) towards the signal, PA modifies them with informative directions – called patterns — that it learns from data.

We show that PA, in turn, suffers from problems that IG has overcome. In particular, it suffers from the saturation problem which occurs at function plateaus (cf. vanishing gradient), resulting in zero-attributions for input features that contributed to non-zero output activations.

*Equal contribution [1]German Research Center for Artificial Intelligence (DFKI). Correspondence to: Robert Schwarzenberg <robert.schwarzenberg@dfki.de>.

2020 ICML Workshop on Human Interpretability in Machine Learning (WHI 2020). Copyright 2020 by the author(s).

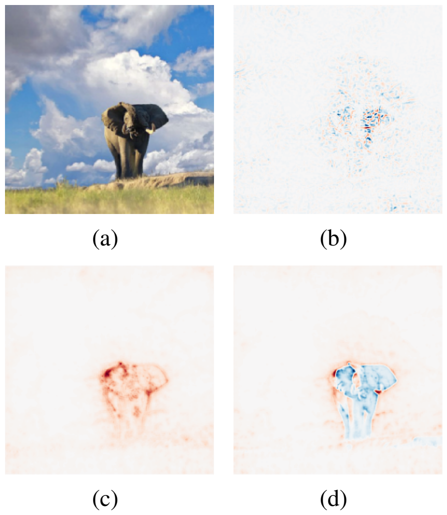

Figure 1. Saliency maps (red: positive, blue: negative) for an input (a) according to IG (b), PA (c), and PGIG (d), explaining the correct (`African Elephant`) VGG-16 classification of (a).

In response, we propose a hybrid approach, Pattern-Guided Integrated Gradients, that combines the strengths of both methods. We demonstrate that PGIG passes controlled stress tests that both parent methods fail. Furthermore, we find that it outperforms its parent methods, as well as seven other prominent explainability methods, in a large-scale image degradation experiment. The new method can be implemented quickly, in particular when Integrated Gradients and PatternAttribution are already part of the code base, as is the case in several explainability frameworks (Wang, 2018; Alber et al., 2019).

2. Prerequisites

In this section, we briefly discuss Integrated Gradients and PatternAttribution to introduce notation as well as important concepts and properties of the two attribution methods. We consider an attribution method a function $\phi_{f,i}(x) : \mathbb{R}^D \rightarrow \mathbb{R}$ that maps each input feature $i \in \{1 \ldots D\}$ in $x \in \mathbb{R}^D$ to a real number that signifies the importance of

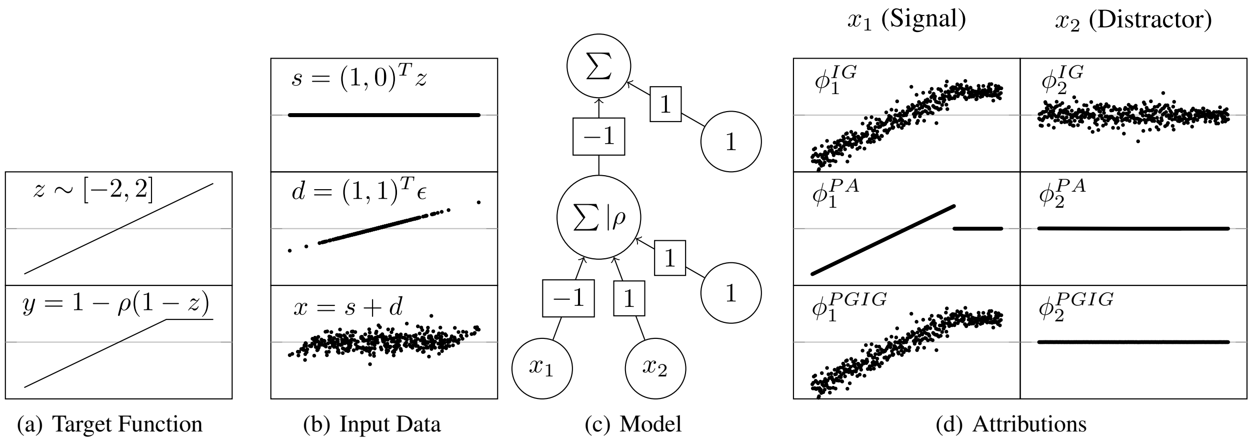

Figure 2. Stress tests for attribution methods, involving a plateau and noise, combining challenges from Sundararajan et al. (2017) and Kindermans et al. (2018). **a** shows the target function (ρ is shorthand for ReLU, horizontal line located at zero). **b** visualizes synthetic input training data x (**b**, bottom row). The data is composed of a signal s (**b**, top row) that introduces information about the target to the first dimension of x and a distractor d (**b**, middle row) that introduces noise to both dimensions of x. Note that the second dimension of x does not carry information about y, just noise. **c** depicts a model (weights in rectangles) which has learned the mapping $x \to y$ by effectively using the signal in x_1 and cancelling the noise from the distractor in x_2. **d** shows attribution scores for the input features according to Integrated Gradients (**d**, top row), PatternAttribution (**d**, middle row) and the proposed Pattern-Guided Integrated Gradients (**d**, bottom row). Only the proposed method attributes importance to the signal in x_1 at the *non-zero* plateau (despite a zero-gradient) and bypasses the noise from the distractor in x_2.

input i to the output of model $f : \mathbb{R}^D \to \mathbb{R}$.[1]

2.1. Integrated Gradients

The attributions provided by Integrated Gradients are a summation of the gradient attribution maps at values from the straight-line path between a baseline $\bar{x}$ (a user-defined reference point) and the input, x. The formula is given by

$$\phi_{f,i}^{IG}(x) = \frac{x_i - \bar{x}_i}{m} \sum_{k=1}^{m} \frac{\partial f(\bar{x} + \frac{k}{m}(x - \bar{x}))}{\partial x_i} \tag{1}$$

where m is a hyperparameter, the number of equidistant steps along the path. As a path method, IG mitigates the aforementioned saturation problem, which we demonstrate below. Furthermore, the authors of Integrated Gradients cite two desirable properties for attribution methods: The first is referred to as *Sensitivity*. An attribution method is *sensitive* “if for every input and baseline that differ in one feature but have different predictions then the differing feature should be given a non-zero attribution” (Sundararajan et al., 2017). The second property is called *Implementation Invariance* and demands that two networks that are functionally equivalent – regardless of implementation – should always yield identical attribution maps. IG is both sensitive and implementation invariant as $\lim m \to \infty$.

[1]For simplicity, without the loss of generality, we assume one-dimensional outputs throughout this paper.

2.2. PatternAttribution

The authors of PatternAttribution (Kindermans et al., 2018) criticize Integrated Gradients, among other gradient methods, for not discriminating signal and distractor in the input data. Their argument is based on the observation that a well-trained model cancels the distractor in the input – that is, everything that it did not find to co-vary with the target. For example, assume that we want to model a simple linear relation $x \to y$, where $x = s + d$ is composed of a signal s that carries all the information needed to predict the target y (it co-varies with the target) and an additive distractor d. In case of a well-trained linear model, the model must have learned weights w s.t. $f(d) = w^T d = 0$ and $f(x) = w^T x = w^T s = y$.

Thus, the weights function as a filter that must always change direction with the distractor in order to stay orthogonal to it. A change in the signal, on the other hand, is accounted for by a change in magnitude of the weights. The authors conclude that gradient methods – including IG – that channel attributions along w inherently direct it towards a direction that is determined by the distractor, not the signal.

For PatternAttribution, prior to the backward pass, the weights w of a linear or ReLU activated layer are replaced by $w \odot p$ where

$$p = \frac{\mathbb{E}_+[\mathbf{x}\hat{\mathbf{y}}] - \mathbb{E}_+[\mathbf{x}]\mathbb{E}[\hat{\mathbf{y}}]}{w^T\mathbb{E}_+[\mathbf{x}\hat{\mathbf{y}}] - w^T\mathbb{E}_+[\mathbf{x}]\mathbb{E}[\hat{\mathbf{y}}]} \tag{2}$$

is a pattern computed over a batch of layer inputs and outputs $\mathbf{x}, \hat{\mathbf{y}}$. $\mathbb{E}_+[\cdot]$ denotes the expectation tensor over the positive regime of a ReLU activated layer, $\{x|w^T x > 0\}$. We can interpret Eq. 2 as follows: Weights that primarily cancel the distractor are scaled down whereas weights that amplify or conserve the signal are preserved. This way, PatternAttribution directs a modified gradient towards the signal. Subsequently, we denote a gradient backward call with the patterns in place as $\partial^{(p)}$. According to this notation, PatternAttribution becomes

$$\phi^{\text{PA}}_{f,i}(x) = \frac{\partial^{(p)} f(x)}{\partial^{(p)} x_i} \tag{3}$$

3. Stress Tests

The authors of Integrated Gradients and the authors of PatternAttribution each present a different stress test to demonstrate the benefits of their method over alternative approaches. The former demonstrate that IG mitigates the saturation problem, the latter prove that PA is able to avoid noise.

We combine the two stress tests by defining a target function that involves a plateau and training a network to model the function with noisy input data. We demonstrate that each method fails the other's test: PA starves at the plateau and IG attributes importance to the noise in the input. We then combine the two methods into Pattern-Guided Integrated Gradients and demonstrate that the hybrid approach passes all tests. We illustrate this argument in Fig. 2.

3.1. Target Function

With IG, PA and PGIG we will later explain a neural network that models $y = 1 - ReLU(1 - z)$, for $z \in [z^{(1)} = -2, z^{(2)} = -1.99, \ldots, z^{(N)} = 2]$, the target function, depicted in Fig. 2 (**a**). Please note that y plateaus after $z > 1$. This *non-zero* plateau is the first stress test for the gradient-based attribution methods because the gradient becomes zero at the plateau but the attribution scores should not be zero.

3.2. Input Data

Let us now generate two-dimensional input training data, $x \in \mathbb{R}^2$ where $x = s + d$. As mentioned above, we want the signal s to co-vary with the target. To generate such a signal, we scale $(1, 0)^T$ with z, s.t. $\mathbf{s} = [(1, 0)^T z^{(1)}, (1, 0)^T z^{(2)}, ...]$. The signal is visualized in Fig. 2 (**b**), top row.

The distractor (Fig. 2 (**b**), middle row) is $d = (1, 1)^T \epsilon$ where $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$, which we sample independently for each $s \in \mathbf{s}$. Note that while s carries information about z and y in the first dimension, d contains only noise, i.e. it does not contain information about the target. Because only d is present in the second dimension, let us subsequently refer to the second dimension as the *distractor dimension* and the first dimension as the *signal dimension*.

3.3. Model

To produce the input given the target, the network must effectively cancel the distractor (below, we will construct such a network). This is the second stress test: If the model cancels the distractor, inputs pertaining only to the distractor should receive only zero attributions. In our case, this means that x_2 should receive only zero attributions, as it contains nothing but noise from the distractor. This is challenging since the respective inputs and weights might not be zero.

Let us first consider a proxy model which learns w such that $w^T x = z$. We can solve for w analytically: Since w needs to be orthogonal to the distractor base vector which is $(1, 1)^T$, $w = (1, -1)^T$. If, for example, $s^T = (.5, 0)$ and $d^T = (.1, .1)$, then indeed $w^T(s + d) = .5 = z = w^T s$ and $w^T d = 0$. Thus, w successfully cancels the distractor.

Now, assume that $f^{(1)}$ and $f^{(2)}$ are two dense layers, with unit biases and parameters $w^{(1)}, w^{(2)}$, accepting two- and one- dimensional inputs, respectively. If we set $w^{(1)} = -w$ and $w^{(2)} = (-1)$ then $y = f(x) = f^{(1)}(\rho(f^{(2)}(x)))$, where ρ is shorthand for ReLU. This model is outlined in Fig. 2 (**c**).

3.4. Attributions

At this point, we have combined the two stress tests from Sundararajan et al. (2017) and Kindermans et al. (2018) and apply IG and PA to receive attributions for $\hat{y}$, depicted in Fig 2 (**d**). For PA, we compute the patterns for $w^{(1)}$ and $w^{(2)}$ with Eq. 2, which yields $p^{(1)} \approx (-1, 0)^T$ and $p^{(2)} = (-1)$. The bias contributions are considered zero, as the bias does not co-vary with the target (Kindermans et al., 2018). For PA, the backpropagation is started with $\hat{y}$, whereas for IG, the backpropagation is invoked starting with 1.0.[2]

IG (Fig. 2 (**d**), top row) follows the function in the signal dimension (x_1) including the plateau after $z > 1$, which we consider desirable. It does, however, attribute a significant portion of importance to inputs from the distractor dimension, which only carries noise and is cancelled by the model. PA (Fig. 2 (**d**), middle row) successfully avoids the noise, i.e. its attribution scores in the distractor dimension are low, but it suffers from the saturation problem at the plateau in the signal dimension due to a zero-gradient. As

[2]For demonstration purposes, we violate the constraint for IG that output values are in the range $[0, 1]$. In our case, this only scales the attributions and does not corrupt the method.

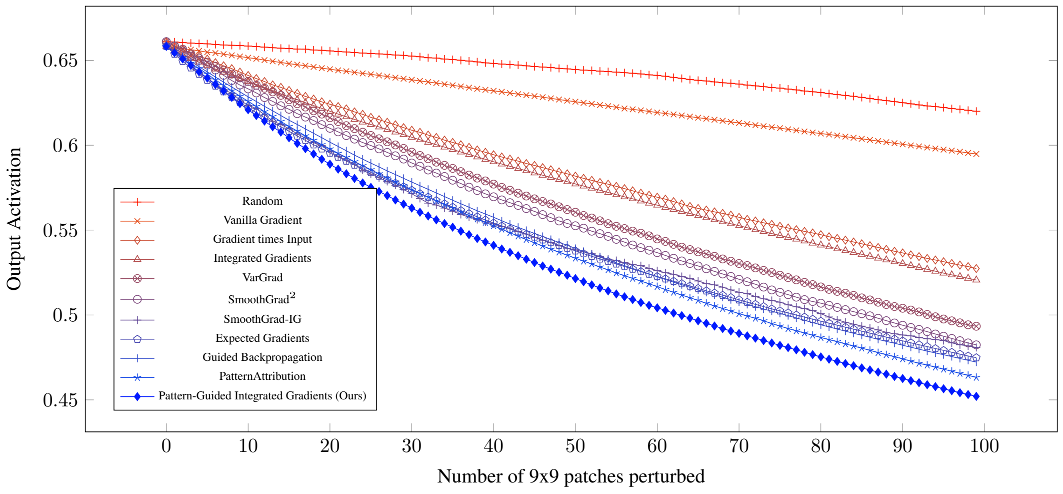

Figure 3. Image degradation experiment using VGG-16 on the val. split of ImageNet (steeper is better).

a consequence, PA violates sensitivity.

4. Pattern-Guided Integrated Gradients

In response, we propose Pattern-Guided Integrated Gradients, given by

$$\phi_{f,i}^{PGIG}(x) = \frac{x_i - \bar{x}_i}{m} \sum_{k=1}^{m} \frac{\partial^{(p)} f(\bar{x} + \frac{k}{m}(x - \bar{x}))}{\partial^{(p)} x_i} \tag{4}$$

PGIG sums over the saliency maps returned by PA for inputs along the straight path between the baseline and the point of interest, x.

4.1. Properties

Like IG, PGIG is a path method that mitigates the saturation problem and like PA, it considers informative directions and thus avoids the distractor. Its favorable attributions are visualized in the bottom row of Fig. 2 (**d**).

In the linear case, we can extract the pattern from Eq. 4 and recover IG:

$$\begin{aligned}\phi_{f,i}^{PGIG}(x) &= \frac{x_i - \bar{x}_i}{m} \sum_{k=1}^{m} \frac{\partial^{(p)} f(\dots)}{\partial^{(p)} x_i} \\ &= \frac{x_i - \bar{x}_i}{m} \sum_{k=1}^{m} p_i \frac{\partial f(\dots)}{\partial x_i} \\ &= p_i \phi_{f,i}^{IG}(x).\end{aligned}$$

PGIG scales the attribution scores of Integrated Gradients according to the class informativeness of input i. PGIG thus is sensitive to changes in all input dimensions $1 \leq i \leq D$ except for when $p_i = 0$. One interpretation of this is that PGIG is not sensitive to changes in pure distractor dimensions, such as x_2 in Fig. 2 (**c**), (**d**). This reasoning directly translates to intermediate and ReLU-activated layers. Regarding implementation invariance, we leave it to future work to prove or disprove this property for PGIG.

5. Experiments

Sundararajan et al. (2017) motivate IG axiomatically but do not study their method empirically. Kindermans et al. (2018) derive their method axiomatically as well, but they also conduct image degradation experiments, an established metric to estimate the quality of saliency methods.

For the image degradation benchmark, a growing number of patches in the input images are replaced with their mean channel values and the output activation[3] (confidence) of the model is monitored. The order in which patches are perturbed is dictated by the accumulated saliencies of their pixels. The premise of this experiment is that the steeper the drop in confidence, the more accurately the attribution method has identified the most important features.

We, too, benchmark PGIG with the image degradation metric. Like Kindermans et al. (2018), we use a pre-trained VGG-16 model (Simonyan & Zisserman, 2014) to generate saliency maps for the 50k images in the validation split of the ImageNet data set (Deng et al., 2009) that are then used to successively degrade the top 100 patches in descending order and aggregate confidence values. Images were cropped to 224x224 and normalized within $[-1, 1]$. Our code base builds on the PyTorch Visual Attribution framework (Wang, 2018) from where we also received the patterns for VGG-16. Data and code are open source: `https://github.com/DFKI-NLP/pgig`.

[3]We explain networks after the final softmax activation.

We compare the patch ranking by PGIG against a random ordering of the patches as well as against rankings determined by nine other gradient-based explainability methods, which are organized into two classes.

The first class contains gradient aggregation methods of which Integrated Gradients is a member. Vargrad (Adebayo et al., 2018) and SmoothGrad (Smilkov et al., 2017) calculate the variance and mean respectively of the input gradients wrt. randomly perturbed inputs. For our experiments, we use the squared version of SmoothGrad (Hooker et al., 2019) as it outperformed its rooted counterpart. Smilkov et al. (2017) also suggest an extension of Integrated Gradients, which merges Integrated Gradients with SmoothGrad, denoted by SmoothGrad-IG. Expected Gradients (Erion et al., 2019) is another derivative of Integrated Gradients that uses baselines drawn from the data distribution to aggregate values.

The second class consists of modifications to the Vanilla Gradient method (Simonyan et al., 2014), of which PatternAttribution is a member. Gradient times Input (Shrikumar et al., 2016) simply multiplies the gradient saliency map by the input, while Guided Backpropagation (Springenberg et al., 2014) aims to filter for positive class evidence by inhibiting negative gradient values as well as those corresponding to negative values in the layer inputs.

6. Results & Discussion

Saliency maps produced by IG, PA and PGIG are shown in Fig. 1. For comparability, we choose the same input image that Kindermans et al. (2018) discuss in their paper. We observe that the heat map generated by PGIG appears plausible, as do the heatmaps of its parent methods: the most salient input features are located in the proximity of the African elephant in the input image.

Regarding faithfulness, confidence curves are plotted in Fig. 3. We observe that, according to the image degradation metric, the random patch ordering performs worst with VGG-16 on ImageNet, as expected. The random ordering is followed by the simple gradient method. It should be mentioned, however, that the simple gradient is more a sensitivity detector than an attribution method.

Gradient times Input and Integrated Gradients both multiply gradients with inputs and perform similarly in our experiment. This is on par with the finding that in the linear case, Input times Gradient and Integrated Gradients even are equivalent methods (Adebayo et al., 2018).

Both methods are surpassed by VarGrad, which itself is exceeded by SmoothGrad^2, SmoothGrad-IG, Expected Gradients, Guided Backpropagation and PatternAttribution; all of which perform similarly. Interestingly, these methods are of different classes: VarGrad, SmoothGrad, SmoothGrad-IG and Expected Gradients are gradient aggregating methods, whereas Guided Backpropagation and PatternAttribution are gradient modifying methods. Thus, we do not see any class membership preference.

Pattern-Guided Integrated Gradients is both, a gradient aggregating method and a gradient modifying method. According to the image degradation metric, it outperforms all other methods tested. However, this result is definitive only for methods without hyper parameters, such as Guided Backpropagation or PatternAttribution. We report the hyper parameters we use in the appendix.

7. Related Work

Much of the related work that inspired the new method has already been mentioned in the previous sections. PGIG is of course based on its parent methods, IG (Sundararajan et al., 2017) and PA (Kindermans et al., 2018). PGIG is not the first method to extend IG, however. Expected Gradients (Erion et al., 2019) and a layered version of Intergrated Gradients (Mudrakarta et al., 2018) are other examples. Integrated-Gradient Optimized Saliency (Qi et al., 2019) uses IG with mask optimization to generate attributions. Unlike PGIG, however, none of these methods apply informative directions.

PGIG is both a modification to the Vanilla Gradient method (Simonyan et al., 2014), such as Guided Backpropagation (Springenberg et al., 2014), and a gradient aggregate method, such as SmoothGrad (Smilkov et al., 2017), VarGrad (Adebayo et al., 2018), or the very recent SmoothTaylor method (Goh et al., 2020). For SmoothTaylor, Goh et al. (2020) bridge IG and SmoothGrad – loosely related to what Smilkov et al. (2017) propose for SmoothGrad-IG – but within a Taylor's theorem framework.

8. Conclusion & Future Work

We present Pattern-Guided Integrated Gradients, which combines Integrated Gradients with PatternAttribution. Due to favorable properties, the new method passes stress tests that both parent methods fail. Furthermore, in a large-scale image degradation experiment, PGIG outperforms nine alternative methods, including its parent methods.

The image degradation metric that we offer to empirically validate the new method is being discussed, however (Hooker et al., 2019). In the future, the new method should thus also be tested against other metrics. Furthermore, IG was shown to have a problematic degree of invariance to model randomization (Adebayo et al., 2018). It should be explored to what degree PGIG still exhibits this behaviour.

Acknowledgements

We would like to thank Christoph Alt, David Harbecke, Moritz Augustin and Arne Binder for their helpful feedback. Furthermore, we would like to thank Jonas Mikkelsen for helping with the code review. This work has been supported by the German Federal Ministry of Education and Research as part of the project BBDC2 (01IS18025E).

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pp. 9505–9515, 2018.

Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K. T., Montavon, G., Samek, W., Müller, K.-R., Dähne, S., and Kindermans, P.-J. iNNvestigate neural networks. *Journal of Machine Learning Research*, 20(93): 1–8, 2019.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Erion, G., Janizek, J. D., Sturmfels, P., Lundberg, S., and Lee, S.-I. Learning explainable models using attribution priors. *arXiv preprint arXiv:1906.10670*, 2019.

Goh, G. S., Lapuschkin, S., Weber, L., Samek, W., and Binder, A. Understanding integrated gradients with smoothtaylor for deep neural network attribution. *arXiv preprint arXiv:2004.10484*, 2020.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 9734–9745, 2019.

Kindermans, P.-J., Schütt, K. T., Alber, M., Müller, K.-R., Erhan, D., Kim, B., and Dähne, S. Learning how to explain neural networks: PatternNet and PatternAttribution. In *International Conference on Learning Representations*, 2018.

Mudrakarta, P. K., Taly, A., Sundararajan, M., and Dhamdhere, K. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1896–1906, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Qi, Z., Khorram, S., and Li, F. Visualizing deep networks by optimizing with integrated gradients. In *CVPR Workshops*, volume 2, 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.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2014.

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

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3319–3328. JMLR. org, 2017.

Wang, Y. Pytorch-visual-attribution. `https://github.com/yulongwang12/visual-attribution`, 2018.

A. Hyperparameters

Below, we cite the formulas of the methods that involve hyper parameters and report the hyper parameters we use in our experiments.

A.1. (Pattern-Guided) Integrated Gradients

Integrated Gradients (Sundararajan et al., 2017) is given by

$$\phi_{f,i}(x) = \frac{x_i - \bar{x}_i}{m} \sum_{k=1}^{m} \frac{\partial f(\bar{x} + \frac{k}{m}(x - \bar{x}))}{\partial x_i}$$

In our experiments, $\bar{x} = 0, m = 25$. For the proposed pattern-guided version we use the same hyperparameters.

A.2. SmoothGrad^2

SmoothGrad^2 (Hooker et al., 2019) is given by

$$\phi_{f,i}(x) = \frac{1}{n}\sum_{1}^{n}\left(\frac{\partial f(x')}{\partial x_i'}\right)^2$$

where $x' = x + \mathcal{N}(\mu, \sigma^2)$ is sampled for each n. In our experiments, $n = 25, \mu = 0, \sigma^2 = 0.15$.

A.3. SmoothGrad-IG

SmoothGrad-IG (Smilkov et al., 2017) is given by

$$\phi_{f,i}(x) = \frac{x_i - \bar{x}_i}{m}\sum_{k=1}^{m}\frac{1}{n}\sum_{1}^{n}\frac{\partial f(\bar{x} + \frac{k}{m}(x' - \bar{x})))}{\partial x_i'}$$

where x' is defined above. In our experiments, $\bar{x} = 0, m = 25, n = 25, \mu = 0, \sigma^2 = 0.15$.

A.4. VarGrad

VarGrad (Hooker et al., 2019) is given by

$$\phi_{f,i}(x) = var_n(\frac{\partial f(x')}{\partial x_i'})$$

where x' is defined above. In our experiments, $n = 25, \mu = 0, \sigma^2 = 0.15$.

A.5. Expected Gradients

Expected Gradients (Erion et al., 2019) is given by

$$\phi_{f,i}(x) = \mathop{\mathbb{E}}_{\bar{x}\sim\mathbf{x}, \alpha\sim U(0,1)}\left[(x_i - \bar{x}_i)\frac{\partial f(\bar{x} + \alpha(x - \bar{x}))}{\partial x_i}\right]$$

In our experiments, we sampled $\bar{x}$ 49 times from the data.