Riemann Sum Optimization for Accurate Integrated Gradients Computation

Swadesh Swain*
Indian Institute of Technology, Roorkee
swadesh_s@ece.iitr.ac.in

Shree Singhi*
Indian Institute of Technology, Roorkee
shree_s@mfs.iitr.ac.in

Abstract

Integrated Gradients (IG) is a widely used algorithm for attributing the outputs of a deep neural network to its input features. Due to the absence of closed-form integrals for deep learning models, inaccurate Riemann Sum approximations are used to calculate IG. This often introduces undesirable errors in the form of high levels of noise, leading to false insights in the model's decision-making process. We introduce a framework, **RIEMANNOPT**, that minimizes these errors by optimizing the sample point selection for the Riemann Sum. Our algorithm is highly versatile and applicable to IG as well as its derivatives like Blur IG and Guided IG. RIEMANNOPT achieves up to 20% improvement in Insertion Scores. Additionally, it enables its users to curtail computational costs by up to four folds, thereby making it highly functional for constrained environments.

1 Introduction

Deep Neural Network (DNN) classifiers for computer vision are increasingly being utilized in critical fields such as healthcare [4] and autonomous driving [3]. Hence, it has become increasingly important to understand the decision-making process for these models. This has led to a growing body of research focused on understanding how the predictions of these deep networks can be attributed to specific regions of the image. An attribution method attempts to explain which inputs the model considers to be most important for its outputs. Several gradient-based [20, 18, 19, 9, 17, 12] and gradient-free [11, 14, 23, 5, 7, 30, 22, 29] attribution methods have been developed for deep learning models. Integrated Gradient methods [27, 10, 24] are a specific class of gradient-based attribution methods that compute a line integral of the gradients of the model over a path defined from a baseline image to the given input.

The complex functional space of deep learning models is often considered as a source of noise for many gradient-based attribution methods, resulting in undesirable high attribution to some background regions. For Integrated Gradients, Kapishnikov et al. [2021] claim that the source of noise is large gradients in the model surface while Smilkov et al. [2017] argue that the source of error is the rapid fluctuation of the gradients of deep learning models.

Deep learning models do not have closed-form integrals, so their integrals are approximated by the Riemann Sums [24]. This approximation involves the sampling of a number of points along the path and approximating the integral using interpolation between these points. Using more points to approximate the Riemann sum naturally results in cleaner saliency maps. However, most applications of Integrated Gradients require a high number of steps for the Riemann Sum [24, 16], generally between 20 to 1000, rendering Integrated Gradients' usage practically unfeasible in real-time applications. On the other hand, using lesser number of samples severely impacts the quality of the saliency map. This results in a trade-off between speed and performance.

*Equal Contribution

Interpretable AI: Past, Present and Future Workshop at NeurIPS 2024

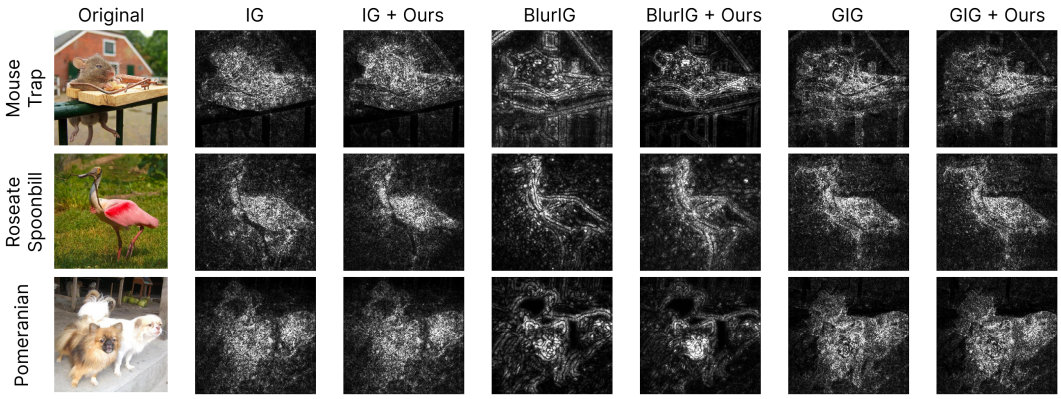

Figure 1: Visual comparison of Integrated Gradient methods with and without RIEMANNOPT. For IG, RIEMANNOPT suppresses the noise around the Spoonbill and also slightly concentrates stronger attribution scores on the mouse trap. Applying RIEMANNOPT to BlurIG significantly increases concentration on the subjects of images. GIG saliency maps remain perceptually similar.

Sotoudeh and Thakur [2019] have attempted to tackle the above issue of inaccurate Integrated Gradients computation by exactly computing the underlying integral using EXACTLINE. However, their application is limited to neural networks that are composed of piece-wise linear operations. Most traditional models, like InceptionV3 [25], ViT [6] and ResNet [8], while being primarily composed of linear operations, also make use of non-linear operations like LayerNorm [1], GroupNorm [26] and Attention [2] thus prohibiting the use of EXACTLINE. Furthermore, the use of EXACTLINE might not be considered ideal in cases where there are computational constraints since it requires ~ 14000 gradient computations per image for large models.

To overcome the problem of redundancy by ineffective sampling schedules prevalent in Integrated Gradient methods, we introduce **RIEMANNOPT**, a framework to pre-determine optimal points for sampling to calculate Riemann Sums. The pre-determined points are specific only to the model. Hence, the computation to determine the points is only done once and does not need to be repeated for every image. Unlike EXACTLINE, our method does not impose any architectural constraints on the underlying model, with the additional benefit of requiring far fewer samples. We present qualitative and quantitative results for RIEMANNOPT on Integrated Gradients (IG) [24], Blur Integrated Gradients (BlurIG) [27] and Guided IG (GIG) [10]. Our method can be easily combined with existing IG-based methods and enable them to generate cleaner saliency maps.

2 Background

In this section, we review the mathematical definition of IG [24], BlurIG [27], and GIG [10].

2.1 Integrated Gradients

Sundararajan et al. [2017] utilized the idea of a path function. $\gamma : [0, 1] \to \mathbb{R}^n$ is a smooth function that denotes a path within $\mathbb{R}^n$ from x' to x, satisfying $\gamma(0) = x'$ and $\gamma(1) = x$. Further, they defined path integrated gradients along the ith dimension for an input x, given a baseline x', obtained by integrating the gradients along the path $\gamma(\alpha)$ for $\alpha \in [0, 1]$ as:

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

Where f denotes a DNN classifier. Integrated Gradients (IG) Sundararajan et al. [2017] originally defined the path method as a straight line path specified $\gamma^{IG}(\alpha) = x' + \alpha \times (x - x') \quad \text{for} \quad \alpha \in [0, 1]$. Later, BlurIG and GIG introduced non-linear paths that had their respective advantages over IG.

2.2 Blur Integrated Gradients

Xu et al. [2020] introduced Blur Integrated Gradients: For a given function $f : \mathbb{R}^{m \times n} \rightarrow [0, 1]$ representing a classifier, let $z(x, y)$ be the 2D input. Blur IG's path is defined by a Gaussian filter that progressively blurs the input. Formally:

$$\gamma^{BlurIG}(x, y, \alpha) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \frac{1}{\pi\alpha} e^{-\frac{x^2+y^2}{\alpha}} z(x - m, y - n) \tag{2}$$

The final BlurIG computation is as follows:

$$I^{BlurIG}(x, y) ::= \int_{\infty}^{0} \frac{\partial f_c(\gamma^{BlurIG}(x, y, \alpha))}{\partial \gamma^{BlurIG}(x, y, \alpha)} \frac{\partial \gamma^{BlurIG}(x, y, \alpha)}{\partial \alpha} d\alpha \tag{3}$$

2.3 Guided Integrated Gradients

Guided IG [10] (GIG) follows an adaptive integration path $\gamma^{IG}(\alpha), \alpha \in [0, 1]$ to avoid high gradient regions. An adaptive path is one that depends on the model being used:

$$\gamma^{GIG} = \underset{\gamma \in \Gamma}{\operatorname{argmin}} \sum_{i=1}^{N} \int_{0}^{1} |\frac{\partial f(\gamma(\alpha))}{\partial \gamma_i(\alpha)} \frac{\partial \gamma_i(\alpha)}{\partial \alpha}| d\alpha, \tag{4}$$

After finding the optimal path γ^{GIG}, GIG computes the attribution values similar to IG,

$$I_i^{GIG}(x) = \int_{0}^{1} \frac{\partial f(\gamma^{GIG}(\alpha))}{\partial \gamma_i^{GIG}(\alpha)} \frac{\partial \gamma_i^{GIG}(\alpha)}{\partial \alpha} d\alpha. \tag{5}$$

3 Methodology

In this section, we present simple derivation to determine an upper bound on the error introduced by approximating a one-dimensional integral using a Riemann Sum. We then extend the definition for multi-dimensional line integrals and define the algorithm RIEMANNOPT uses to schedule samples to minimize this upper bound.

3.1 Error Minimization of Riemann Sums in 1D

We now present the derivation to estimate the error introduced due to the left Riemann Sum approximation of a standard 1D integral $\int_{\alpha_0}^{\alpha_k} g(\alpha)\, d\alpha$ where $\{\alpha_i\}_{i=0}^{k}$ is the set of points at which the integrand, $g(\alpha)$, is evaluated.

The standard way to calculate the left Riemann Sum is:

$$R = \sum_{i=0}^{n-1} g(\alpha_i)(\alpha_{i+1} - \alpha_i) \tag{6}$$

The integral can be broken down as:

$$I = \sum_{i=0}^{n-1} \int_{\alpha_i}^{\alpha_{i+1}} g(\alpha) d\alpha \tag{7}$$

By applying the Taylor Series approximation around α_i in (7):

$$
\begin{aligned}
I &\approx \sum_{i=0}^{n-1} \int_{\alpha_i}^{\alpha_{i+1}} g(\alpha_i) + (\alpha - \alpha_i) g'(\alpha_i)\, dx \\
&\approx \sum_{i=0}^{n-1} g(\alpha_i)(\alpha_{i+1} - \alpha_i) + g'(\alpha_i) \frac{(\alpha_{i+1} - \alpha_i)^2}{2}
\end{aligned} \tag{8}
$$

By (6), (8) and the Triangle Inequality:

$$
|R - I| \lesssim \frac{1}{2} \sum_{i=0}^{n-1} \left| g'(\alpha_i)(\alpha_{i+1} - \alpha_i)^2 \right| \tag{9}
$$

3.2 Algorithm

IG computes d integrals (attributions) per image, one for each pixel. We treat each integral independently and use the derivation above to estimate the average error over all integrals. The input to the function, g, would be multidimensional, resulting in a different Taylor Series expansion. However, the approximation would still be mathematically sound since the integral corresponding to the ith feature is only dependant on the gradient along that component, i.e. error corresponding to the ith dimension of gradient only contributes to the ith integral. We use this observation in conjunction with the finite distance approximation of the derivative to determine the optimal points for sampling a Riemann Sum for the dataset. The primary idea behind the algorithm is to approximate the average $|g'(\alpha)|$ for all input features on a small subset of images, $\sim 1\%$ of the validation dataset, then compute the optimal sampling points and use them for the entire dataset.

The following tensors are used in Algorithm 1 where d is the dimensionality of the input:

- $I_{k \times d}$: Samples evaluated at k equispaced points along the path.
- $C_{k-1 \times d}$: Finite difference estimate of the derivative of I for all input features.
- A_{k-1}: Absolute derivative estimate of I, corresponding to $|g'(\alpha)|$

Algorithm 1 Estimation of Optimal Alphas

Inputs:
A subset of m examples from the validation dataset: $X_i \in \mathbb{R}^d, i \in \{1, \ldots, m\}$
Number of sample points in a path: k
Integrand of the IG method: $\frac{\partial f(\gamma(\alpha))}{\partial \gamma(\alpha)} \odot \frac{\partial \gamma(\alpha)}{\partial \alpha}$
Output:
Optimal sampling points: $\{\alpha_j^*\}_{j=1}^k$
Initialization:
Set $\{\alpha_j\}_{j=1}^k$ as k linearly spaced scalars between the integral bounds
$A \leftarrow$ Initialize with zeros
for each i **in** $\{1, \ldots, m\}$ **do** ▷ Loop over training examples
$I_j \leftarrow \frac{\partial f(\gamma(\alpha))}{\partial \gamma(\alpha)} \odot \frac{\partial \gamma(\alpha)}{\partial \alpha}\Big|_{\alpha=\alpha_j}$ for j in $\{1, \ldots, k\}$
$C_{k-1 \times d} \leftarrow \frac{I_{j+1} - I_j}{\alpha_{j+1} - \alpha_j}$ for j in $\{1, \ldots, k-1\}$ ▷ Finite difference: $g'(\alpha) \approx \frac{g(\alpha + \Delta\alpha) - g(\alpha)}{\Delta\alpha}$
Apply element-wise absolute to C
A_{k-1} += Average $C_{k-1 \times d}$ across all features ▷ Estimate of $|g'(\alpha)|$
end for
Normalize A by dividing by number of examples m
$|g'(\alpha)| \leftarrow$ Linearly Interpolate($A,\ \alpha$) ▷ $\alpha \in [0, 1], A \in \mathbb{R}^{k-1}$
$\alpha_j^* \leftarrow$ The set $\{\alpha_j\}_{j=1}^k$ that minimizes the upper bound error defined by Equation (9)
return $\{\alpha_j^*\}_{j=1}^k$

4 Experimental Setup and Metrics

In this section, we discuss the details of the implementation, dataset, model, and metrics used.

4.1 Experimental Setup

We use the original implementations with default parameters in the authors' code for IG, GIG, and BlurIG and implement RIEMANNOPT as a pre-computation step that links with the original implementations. We present our results using InceptionV3 for 16, 32, 64 and 128 sample points on the correctly classified images of the ImageNet validation dataset, $\sim 40K$. To estimate $|g'(\alpha)|$, we apply Algorithm 1 to a set of 200 randomly correctly classified images from the ImageNet validation dataset for 128 samples. Then, we use Powell's method [15] to determine the optimal set of sampling points. This roughly has the same computational cost as computing the saliency map for the set of 200 images. Using RIEMANNOPT is still cost-effective since we only use a small number of images to calculate sample points but are able to use these points for the entire dataset.

4.2 Metrics

Previous works use the Insertion Score and Normalized Insertion Score to compare different attribution methods [27, 9, 10, 28, 14, 13]. It is critical to note that the purpose of the Insertion Score is to measure the efficacy of a saliency map, i.e. it is not designed to measure how close the Riemann Sum is to the actual integral. However, it is reasonable to assume that the true saliency map would generally achieve better Insertion Scores than an inaccurate approximation since inaccurate estimates introduce noise. Hence, we report the Insertion Scores and, additionally, employ the Axiom of Completeness [24] to define a new metric that measures the quality of the saliency maps without the need for this hypothesis.

According to the Axiom of Completeness, the sum of all feature attributions, determined by any Integrated Gradients method, must ideally add up to the difference between the output of f at x and x'. However, there is always an error due to inaccurate Riemann Sum estimates. Furthermore, Sundararajan et al. [2017] advise the developer to ensure that all feature attributions add up to $f(x) - f(x')$ (within 5%) and suggest increasing the number of samples if the error is greater.

Since the ground truth is unavailable, it is non-trivial to determine the numerical accuracy of a computed saliency map. We use the relative error between the sum of feature attributions and $f(x) - f(x')$ to estimate the error. This metric is not infallible since the features' positive and negative errors partially offset each other during the summation. Using the Triangle Inequality, it can be easily shown that this metric is a lower bound on the true error. Nevertheless, it serves as a helpful proxy since near-perfect saliency maps will have near-zero error, and highly erroneous maps will, on average, have high error even after the errors partial offset.

5 Results and Discussion

In this section, we compare the sampling points chosen by RIEMANNOPT to the linear schedules, followed by qualitative and quantitative evaluation against the baselines. In the case of BlurIG, the sample points chosen by RIEMANNOPT highly differ from the linearly spaced samples, as depicted in Figure 2. Every path starts with an information-less baseline image, x', and gradually gains perceptible features as it moves towards the input image, x. Along the path, when the image becomes perceptible, the gradients rapidly change, resulting in large values of $|g'(\alpha)|$. For BlurIG, the image features become perceptible at the end of the path when most of the sharpening occurs, $\alpha \approx 0.8$. For IG and GIG, the image becomes perceptible as soon as its brightness crosses a certain threshold, $\alpha \approx 0.1$. The abnormal spike at the end of GIG's curve is due to it's adaptive path mechanism.

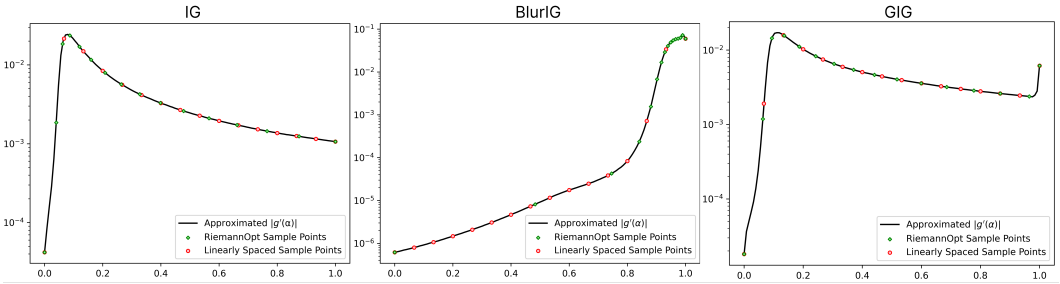

Figure 2: Estimated $|g'(\alpha)|$ and comparison of 16 linearly spaced samples and 16 optimal samples chosen by RIEMANNOPT. High values of $|g'(\alpha)|$ indicate regions of the path where the gradients of the model are rapidly changing, i.e. regions where the image becomes perceptible to the model.

RIEMANNOPT always reduces the relative error and improves metric scores across all methods and sample counts as depicted in Table 1 and Figure 3 respectively, with a noticeable enhancement for BlurIG. On the other hand, the improvement in GIG is not very significant. The path of GIG is theoretically fixed for chosen model. However, due to the employment of an adaptive path, its practical implementation is highly dependent on the number of samples as well as the location of the samples, unlike BlurIG and IG. In the derivation 3.1 of RIEMANNOPT, we assumed that the path function was constant and independent of the sample points. The practical implementation of GIG breaks this assumption; this is a possible explanation for why GIG is not as improved by RIEMANNOPT as the other methods are. In terms of relative error, RIEMANNOPT significantly reduces the number of samples while maintaining comparable performance. Specifically, BlurIG + RIEMANNOPT achieves similar results with 16 samples as BlurIG with 64 samples. Additionally, BlurIG + RIEMANNOPT with 16 samples performs comparably to BlurIG with 32 samples, and GIG + RIEMANNOPT matches the performance of GIG with 128 samples using just 16 samples. This makes RIEMANNOPT highly functional for computationally constrained environments.

The reader may question to what extent the optimal points, approximated for the entire dataset, generalize for each individual image. This can be verified by applying RIEMANNOPT for several individual images and qualitatively inspecting how much the resultant points vary for different images. For ImageNet, we observed that the set of points generated was roughly the same. One would expect the same trend even with different datasets since the characteristics of $|g'(\alpha)|$ are primarily determined by the path construction and model's training procedure rather than the image itself. InceptionV3 was trained with augmentation techniques including random brightness shifts which increased its robustness to dark images, making the image perceptible to the model at around $\alpha \approx 0.1$, as previously mentioned. Future work could investigate how changing the in-training augmentation parameters would affect the shape of the graph and consequently RIEMANNOPT'S chosen points, including blurring augmentations. One would expect the rise in $|g'(\alpha)|$ for BlurIG's path to occur earlier since the model would be better at identifying blurry images, similar to the effect caused by brightness augmentation.

Table 1: Relative Error (↓) across different methods

Method	16 Samples	32 Samples	64 Samples	128 Samples
IG	0.708	0.374	0.166	0.066
IG + RIEMANNOPT	**0.404**	**0.223**	**0.123**	**0.065**
BlurIG	0.886	0.554	0.268	0.114
BlurIG + RIEMANNOPT	**0.269**	**0.123**	**0.058**	**0.041**
GIG	0.786	0.788	0.725	0.612
GIG + RIEMANNOPT	**0.666**	**0.731**	**0.711**	**0.610**

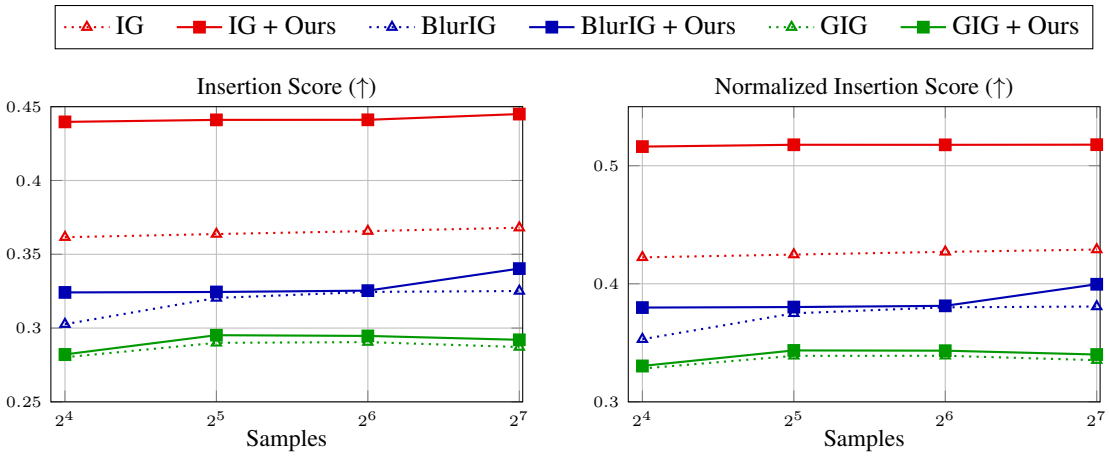

Figure 3: We compare RIEMANNOPT against the baseline methods using the Insertion Score and Normalized Insertion Score. We observe noticeable improvement for BlurIG and IG.

6 Conclusion

In this paper, we present RIEMANNOPT, a highly efficient framework designed to optimize sample points in Riemann Sums for the computation of Integrated Gradients. Both qualitative and quantitative results demonstrate that RIEMANNOPT effectively minimizes numerical errors in saliency maps and achieves improved Insertion Scores by up to 20%, thereby enhancing the accuracy and reliability of attribution maps. RIEMANNOPT is adaptable, extending its applicability to any multi-dimensional line integral computation, including derivatives of Integrated Gradients such as BlurIG and GIG. Additionally, it enables users to curtail computational costs by up to fourfold, significantly boosting efficiency. Opportunities for future work include extending RIEMANNOPT to further improve its suitability for Integrated Gradient methods that employ adaptive paths.

Acknowledgments and Disclosure of Funding

We would like to thank Gaurav Kumar Nayak, Aayan Yadav, Shweta Singh, Anupriya Kumari and Devansh Bhardwaj for their insights on the paper writing. We would also like to thank all members of the Data Science Group of IIT Roorkee for their invaluable support.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *ArXiv*, 2014.

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

Bresnick G Cuadros J. Eyepacs: An adaptable telemedicine system for diabetic retinopathy screening. In *Journal of Diabetes Science and Technology*, 2009.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit,

and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viegas, and Michael Terry. Xrai: Better attributions through regions. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Andrei Kapishnikov, Ben Wedin, Besim Namik Avci, Michael Terry, Subhashini Venugopalan, 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 (CVPR), 2021*, 2021.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017.

Ettore Mariotti, Jose M. Alonso-Moral, and Albert Gatt. Measuring model understandability by means of shapley additive explanations. In *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2022.

Deng Pan, Xin Li, and Dongxiao Zhu. Explaining deep neural network models with adversarial gradient integration. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models, 2018.

M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 1964.

Kristina Preuer, Günter Klambauer, Friedrich Rippmann, Sepp Hochreiter, and Thomas Unterthiner. Interpretable deep learning in drug discovery, 2019.

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*. Association for Computing Machinery, 2016.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

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

Matthew Sotoudeh and Aditya V Thakur. Computing linear restrictions of neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2015.

Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 9269–9278, 2020.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*. PMLR, 2017.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

S. Xu, S. Venugopalan, and M. Sundararajan. Attribution in scale and space. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2020.

Ruo Yang, Binghui Wang, and Mustafa Bilgic. Idgi: A framework to eliminate explanation noise from integrated gradients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis, 2017.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 2013.