One-Step Pixel-Level Perturbation-Based Saliency Detector

Vinnam Kim* MakinaRocks, Republic of Korea
vinnam.kim@makinarocks.ai
Hyunsouk Cho*† Ajou University, Republic of Korea
hyunsuok@ajou.ac.kr
Sehee Chung NCSOFT, Republic of Korea
seheechung@ncsoft.com

Abstract

To explain deep neural networks, many perturbation-based saliency methods are studied in the computer vision domain. However, previous perturbation-based saliency methods require iterative optimization steps or multiple forward propagation steps. In this paper, we propose a new perturbation-based saliency that requires only one backward propagation step by approximating the perturbation effect on the output in the local area. We empirically demonstrate that our method shows fast computations and low memory requirements comparable to other most efficient baselines. Furthermore, our method simultaneously considers all possible perturbing directions so as not to misestimate the perturbation effect. Our ablation study shows that considering all possible perturbing directions is crucial to obtain a correct saliency map. Lastly, our method exhibits competitive performance on the benchmarks in evaluating the pixel-level saliency map. Code is available at https://github.com/vinnamkim/OPPSD.

1 Introduction

Unlike the linear model, which guarantees transparent model behavior to end-users, deep neural networks (DNNs) exhibit opaque behaviors. DNNs are composed of complex nonlinear layers, which are difficult to decompose into interpretable units. In comparison to the superiority of DNNs in solving given challenges, the inherent opaqueness of DNNs has been a hurdle for decades. From an end-user perspective, a model with non-transparency cannot fully obtain trust in its use. This is noticeable when the model is less powerful than humans or has equivalent confidence to humans. In contrast, if the model is transparent, humans can compensate for or reinforce the judgment of the model by using the model transparency (for example, doctors using medical image analysis) [19]. In fields with an abundance of machines (such as playing games [6, 24]), the model transparency also allows the machines to teach humans to make better decisions.

Owing to the difficulty of introducing interpretability into DNNs, the saliency map has been proposed as a tool to shed light on the black-box behaviors of DNNs. The saliency map assigns an importance value to each pixel, which represents the extent to which it contributes to the model output. However, the schemes for scoring the pixel importance value to the output differ in previous studies. One such scheme back-propagates the importance signals from the output to the input [25, 28, 29]. As the representative method, back-propagating gradients from the output to the input is effective

[*] Most work was done in NCSOFT

[†] Corresponding author

given that certain DNN classes represent piece-wise linear functions [3]. Another scheme decomposes the difference in the outputs between the input and the other baseline into the input dimension [23, 30]. Therefore, the obtained importance scores can be summed to represent the output difference precisely. In another approach, the input features are perturbed and the model output changes according to the perturbation are observed [10, 26, 31, 32]. In this scheme, it is expected that the perturbation to the important features affects the model output significantly, but the perturbation to the negligible features has less influence on the model output. The advantage of this scheme over the others is that because it is more intuitively defined, the inevitable ambiguity that occurs when explaining the neural network can be avoided.

However, two major difficulties are encountered in obtaining a pixel-level saliency map using the perturbation schemes. The first issue is the expensive computational cost. Modern DNNs consist of heavily complex structures, and a huge computational cost is required for the forward propagation step. To compose a pixel-level saliency map, each pixel should be perturbed individually to observe the influence on the output. As a result, the perturbation scheme requires $O(H \times W)$ forward propagation steps for a 2D image size of $H \times W$. The other issue is that of misestimating a pixel importance value occurred by not incorporating all perturbation directions. For example, if the object to be classified is mostly dark, perturbation in a uniform negative direction for RGB channels (making pixels darker) will greatly underestimate the effect on the classification output. In other words, it is necessary to consider the sensitivity of changes in the model output that cannot be obtained from a single perturbation vector. However, this problem further complicates the computational cost issue mentioned previously.

According to the above considerations, we propose a novel perturbation-based saliency method that integrates the gradient-based scheme. This means that we can represent the output difference when perturbing the input as a function of the input–output gradients. As a result, we can solve the two problems that arise when applying perturbation schemes to obtain pixel-level saliency maps. To resolve the computational complexity issue, our method replaces multiple forward propagation steps of the model with one backward propagation step. This replacement is made possible by an approximation of the output difference using the first-order Taylor expansion. Furthermore, our method considers all possible directions of the channel-wise perturbation. Unlike the perturbation-based methods that require additional iterative steps to consider all possible directions [26, 32], our method could compute perturbation effects in one step through eigenvalue analysis.

Our contributions are summarized as follows:

- We introduce a new pixel-level saliency method based on the notion of the perturbation scheme. Our method not only requires a small computational cost but also prevents underestimation of the pixel importance value by not relying on one perturbation direction.
- Our method exhibits competitive benchmark performance compared to other pixel-level saliency methods. Owing to the reduction in the computation costs, we evaluate our method using pixel-level benchmarks for the first time, whereas the previous perturbation-based saliency methods never involved pixel-level benchmarks.
- Our ablation studies empirically show that our main idea of eigenvalue analysis for each pixel is effective in estimating the perturbation effect in every direction. Excluding these components from our method, benchmark performances are significantly reduced as intended.

2 Related Works

As mentioned briefly in the previous section, saliency methods can be categorized into three classes. The first class of saliency methods is based on the back-propagation of importance from the output to the input. The most well-known importance back-propagation is using input–output gradients [25]. DeconvNet [31] defines an inversion process for the convolutional neural network to map feature importance. GuidedBackProp [28] propagates only positive gradients backward to mimic the ReLU

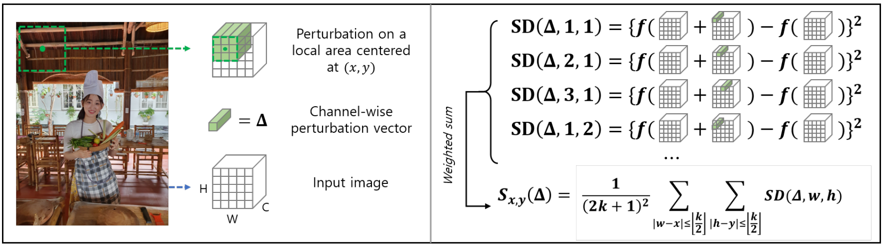

Perturbation on a local area centered at (x, y)
= Δ
Channel-wise perturbation vector
H
W
C
Input image
Weighted sum
SD(Δ, 1, 1) = {f(+) − f( )}²
SD(Δ, 2, 1) = {f(+) − f( )}²
SD(Δ, 3, 1) = {f(+) − f( )}²
SD(Δ, 1, 2) = {f(+) − f( )}²
…
S_{x,y}(Δ) = 1/(2k+1)² Σ_{|w−x|≤⌊k/2⌋} Σ_{|h−y|≤⌊k/2⌋} SD(Δ, w, h)

Figure 1: Channel-wise perturbation on $k \times k$ area centered at (x,y).

activation of the forward-propagation. SmoothGrad [27] and VarGrad [1] add noises to the inputs and gather the gradient statistics to obtain a robust saliency map. FullGrad [29] also includes the importance signals propagated from the bias terms of the neural network, which were ignored in previous studies.

The second class of saliency methods includes those that establish completeness, where the summation of the pixel importance values corresponds to the output difference between the original input and reference input. IntegratedGradient [30], DeepLIFT [23], and deep Taylor decomposition [16] are saliency methods with completeness. Although saliency methods with completeness secure desirable justification for their saliency maps, the existence of the reference input may cause unwilling biases in the importance values.

The third class of saliency methods is based on perturbation to the input, which is the closest to our method. By completely occluding a segment of the input image, the model output changes according to the amount of information removed, which can be utilized as a saliency map [31]. To overcome the limitation of the occlusion method only performing perturbation in a fixed direction, the target patches in the image are sampled conditionally on the other pixels of the broader region nearby. Subsequently, the output differences of the neural network are marginalized over the samples [32]. Another method that considers all possible perturbation directions maximizes the entropy of the Gaussian noise to perturb the input [10]. The other method directly maximizes the loss function of the original classifier with respect to perturbed features [26]. These three approaches are successful in considering all possible perturbation directions, but they still require expensive computation costs for multiple forward-propagation or iterative optimization steps.

To the best of our knowledge, our work is the first to tackle two problems in perturbation-based saliency methods: the consideration of all possible perturbation directions and overcoming the inevitable heavy computational budget. The benchmarks used to report our remarkable results have been broadly accepted in previous studies: the pixel perturbation benchmark [4, 5, 21] and RemOve And Retrain (ROAR) [12].

3 Methodology

3.1 Problem Definition

In this paper, we want to find the sensitivity of the neural network output changes by applying channel-wise perturbation. In Figure 1, we present an example that the neural network output changes by a channel-wise perturbation vector on a $k \times k$ local area centered at (x,y). We define the scalar output of the neural network as $f \in \mathbb{R}^{W \times H \times C} \to \mathbb{R}$, and an input image as a 3D tensor $X \in \mathbb{R}^{W \times H \times C}$, where W is the width, H is the height, and C is the number of channels of the image. From a channel-wise perturbation vector $\mathbf{\Delta} = [\Delta_1, \ldots, \Delta_C]^\top \in \mathbb{R}^C$, we define a channel-wise perturbation tensor

$I(\boldsymbol{\Delta},w,h) \in \mathbb{R}^{W\times H\times C}$ for the pixel (w,h), where $I(\boldsymbol{\Delta},w,h)_{i,j,\cdot} := \boldsymbol{\Delta}$ if $i=w$ and $j=h$; otherwise, $I(\boldsymbol{\Delta},w,h)_{i,j,\cdot} := \mathbf{0}$. Then, $SD(\boldsymbol{\Delta},w,h) := \{f(X+I(\boldsymbol{\Delta},w,h)) - f(X)\}^2$ is the square difference between the original output and perturbed output to the pixel (w,h).

After defining the pixel-level perturbation error $SD(\boldsymbol{\Delta},w,h)$, we collect $SD(\boldsymbol{\Delta},w,h)$ over the area with a size of $k\times k$ pixels that is centered at (x,y) to obtain the perturbation error $S_{x,y}(\boldsymbol{\Delta})$:

$$S_{x,y}(\boldsymbol{\Delta}) := \frac{1}{(2k+1)^2} \sum_{|w-x|\le\lfloor\frac{k}{2}\rfloor} \sum_{|h-y|\le\lfloor\frac{k}{2}\rfloor} SD(\boldsymbol{\Delta},w,h). \tag{1}$$

In previous pixel-level computer vision algorithms, it was common to combine information from neighborhood pixels of the target pixel [11, 15]. This is because an image naturally exhibits spatial correlations between nearby pixels. The locally combined information is less sensitive than the individual pixel information owing to the noise reduction. In the same way, our perturbation error is not only obtained from the single-pixel perturbation, but also by gathering over the area around the target pixel (x,y).

Using Equation 1, we can compute the influence of the channel-wise perturbation $\boldsymbol{\Delta}$ to the local area around the pixel (x,y) on the output difference of the neural network. However, computing Equation 1 is practically expensive because it requires a forward propagation step of the DNN for each perturbation vector $\boldsymbol{\Delta}$ and pixel (x,y). To tackle this problem, in the following two sections, we firstly approximate Equation 1 by the first-order Taylor expansion to avoid the expensive computing cost of a forward propagation step of the DNN. In addition, our actual goal is to obtain the sensitivity of neural network output changes that are not bound to the fixed perturbation vector $\boldsymbol{\Delta}$. In the last section, we propose our method to efficiently incorporate the effects of $\boldsymbol{\Delta}$ in all directions.

3.2 The first-order Taylor Expansion

In this section, we approximate Equation 1 to reduce the computational cost by replacing a forward propagation step of the DNN with a quadratic form. We obtain the first-order Taylor expansion of Equation 1 using the partial derivatives vector $\frac{\partial f(X)}{\partial X_{w,h,\cdot}} := \left[\frac{\partial f(X)}{\partial X_{w,h,1}}, \ldots, \frac{\partial f(X)}{\partial X_{w,h,C}}\right]^\top$:

$$S_{x,y}(\boldsymbol{\Delta}) \approx \frac{1}{(2k+1)^2} \sum_{|w-x|\le\lfloor\frac{k}{2}\rfloor} \sum_{|h-y|\le\lfloor\frac{k}{2}\rfloor} \left(\boldsymbol{\Delta}^\top \frac{\partial f(X)}{\partial X_{w,h,\cdot}}\right)^2 = \boldsymbol{\Delta}^\top \boldsymbol{J}_{x,y} \boldsymbol{\Delta}, \tag{2}$$

where $\boldsymbol{J}_{x,y} := \sum_{|w-x|\le\lfloor\frac{k}{2}\rfloor} \sum_{|h-y|\le\lfloor\frac{k}{2}\rfloor} w_{w,h}(x,y) \frac{\partial f(X)}{\partial X_{w,h,\cdot}} \frac{\partial f(X)}{\partial X_{w,h,\cdot}}^T$.

Deng et al. [8] recently studied the first-order Taylor expansion framework to explain the attribution of DNNs. Although their framework was helpful to give theoretical groundings to the previous saliency methods, their study has limitations in integrating the effects of all directions of Δ. Only setting Δ to a constant vector or sampling Δ from some distributions are discussed in [8]. The sensitivity of neural network output changes could be measured by sampling Δ from some distributions. However, multiple computations are required for each Δ sampled. Instead of sampling Δ, in the next section, we propose a novel approach that considers the perturbation effects in all directions in the equation 2.

3.3 One-Step Pixel-Level Perturbation-Based Saliency Detector

Because Equation 2 is a quadratic form with a column vector Δ and a matrix $\boldsymbol{J}_{x,y}$, it decomposes into the linear combination of 1-dimensional projections of Δ on the eigenvectors. Therefore, we consider the eigenvalues of $\boldsymbol{J}_{x,y}$ in Equation 2 to measure the sensitivity of the perturbation effects in all directions of Δ. The eigenvalues of $\boldsymbol{J}_{x,y}$, λ_i for $i=1,\ldots,C$ represent the sensitivity of $S_{x,y}(\Delta)$ when changing Δ along the direction in the eigenbasis. Instead of sampling Δ, we could measure the sensitivity of the perturbation effects in all directions by incorporating the eigenvalues of $\boldsymbol{J}_{x,y}$ into a representative scalar value.

It is inspired by the well-known corner detection algorithm in the computer vision field [11]. In the Harris corner detection algorithm [11], the two main concepts for extracting corners in a given image are (1) the collection of information over the patch around the target pixel and (2) a corner measure to integrate eigenvalues in the first-order Taylor expansion of the weighted SSD. The first concept is considered in Equation 1 to obtain the weighted SSD over $k \times k$ pixels around the target pixel. Subsequently, we refer to the second concept in this section.

We integrate all eigenvalues into a scalar corner measure to compose the saliency map $\mathbb{M} := \{M_{x,y} \in \mathbb{R}^2 : x = 1,\dots,W,\ y = 1,\dots,H\ \}$. We propose three corner criteria:

- the corner criterion of Noble [17], $M^{\text{noble}}_{x,y} := \frac{\det(\boldsymbol{J})}{\operatorname{trace}(\boldsymbol{J})+\varepsilon}$ for some $\varepsilon > 0$,
- the Frobenius norm, $M^{\text{fro}}_{x,y} := \left\|\boldsymbol{J}_{x,y}\right\|_{\text{F}}$, and
- the minimum eigenvalue, $M^{\min}_{x,y} := \lambda_1$.

These corner criteria consider all directions of the eigenvectors. The corner criterion of Noble [17] represents the harmonic mean of the eigenvalues when $C = 2$. The Frobenius norm is the root sum square of all eigenvalues. The minimum eigenvalue also guarantees that $M^{\min}_{x,y} \leq \lambda_i$ for $i = 1,\dots,C$ because $\boldsymbol{J}_{x,y}$ is a positive semi-definite matrix.

To conclude, our one-step pixel-level perturbation-based saliency detector (OPPSD) firstly obtains the partial derivatives of the DNN output with respect to the input. This procedure could be conducted in only one backward propagation step. Secondly, we compute a corner criterion for each pixel to integrate the sensitivity of the channel-wise perturbation effects. The computational cost for these corner criteria is sufficiently small compared to the backward propagation cost. Finally, the importance score obtained for each pixel represents a cornerness score reflecting the sensitivity of the neural network output change.

4 Experiments

Method	Citation	Multiple backward passes?	Restriction in the model architecture?
OPPSD-Noble	$M^{\text{noble}}_{x,y}$ with a 7×7 window function	$\times$	$\times$
OPPSD-Frobenius	$M^{\text{fro}}_{x,y}$ with a 7×7 window function	$\times$	$\times$
OPPSD-MinEigenvalue	$M^{\min}_{x,y}$ with a 7×7 window function	$\times$	$\times$
FullGrad	Srinivas and Fleuret [29]	$\times$	Only usable for CNN
InputGradient	Simonyan et al. [25]	$\times$	$\times$
IntegratedGradient	Sundararajan et al. [30]	$\surd$	$\times$
VarGrad	Adebayo et al. [1]	$\surd$	$\times$
GuidedBackProp	Springenberg et al. [28]	$\times$	Only usable for ReLU
GuidedGradCam	Selvaraju et al. [22]	$\times$	Only usable for CNN

Table 1: Our methods and all baselines used in our benchmarks.

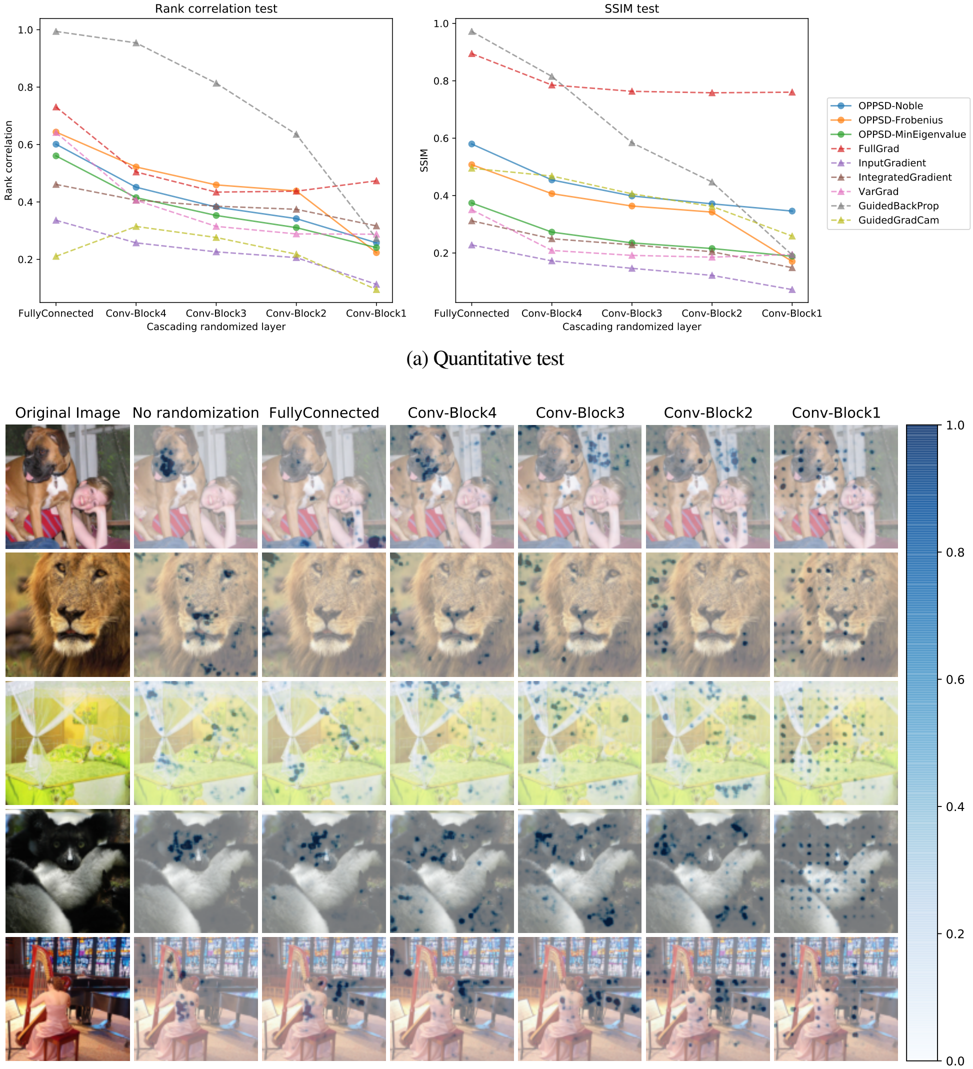
In this section, we compare our methods (OPPSD-*) with the other saliency methods. Our methods and all baselines used in the experiments are listed in Table 1. All baselines were selected because they could afford the computational cost of deriving the pixel-level saliency map. However, some methods were missing in the experiments using VisionTransformer-Base [9] because they have restrictions in the model architecture.

We used all baseline methods implemented by Captum[‡] [13], except for FullGrad[§] [29]. Apart from our methods, all baselines methods produced a sub-pixel-level saliency map. For fairness in

[‡] https://captum.ai

[§] https://github.com/idiap/fullgrad-saliency the comparison, the importance values of the sub-pixel-level saliency map were averaged by channels, following which the absolute values were used to obtain a pixel-level saliency map.

4.1 Sanity Checks



(a) Quantitative test

(b) Qualitative test

Figure 2: Cascading model parameter randomization tests provided with ResNet-50 model for ImageNet dataset. We randomized the model parameters cascadingly at the top fully connected layer and the first convolutional layer of four building blocks of the ResNet-50 model.

Sanity checks proposed by Adebayo et al. [2] are the cascading model parameter randomization tests. The model parameters are randomized cascadingly from the top layer to the bottom layer. If the original model is well-trained, we expect a huge difference between the saliency map of the original model and the saliency map of the model with the randomized parameters. This means that the saliency method correctly explains the well-trained model, rather than exploiting the model-free features of the image; for example, detecting edges.

In Figure 2, we provide two quantitative tests and a qualitative test. Two quantitative tests consist of the rank correlation test and the structural similarity index measure (SSIM) test. In both tests, the divergence between the original saliency map and the saliency map from the model, where parameters are randomized, increases as the metric value approaches zero. Therefore, saliency methods with a curve much greater than zero are hardly accepted as correctly explaining the model behavior. For example, GuidedBackProp remains at approximately 1.0 until the cascading randomization at the convolutional layer of the fourth block for both tests. This result corresponds to the proof of Adebayo et al. [2], whereby GuidedBackProp is insensitive to the model parameter randomization. However, the curves of our methods in both tests significantly drop to zero as soon as the model parameters are randomized. These trends are similar to other saliency methods discussed in [2] as correctly reflecting the model behavior.

Our qualitative test also empirically demonstrates the correctness of our method. Pixels with high importance are highlighted in blue. In Figure 2, as goes to the right column, the pixels that were important in the second column (no parameter randomization) are randomly scattered.

4.2 Pixel Perturbation Benchmark

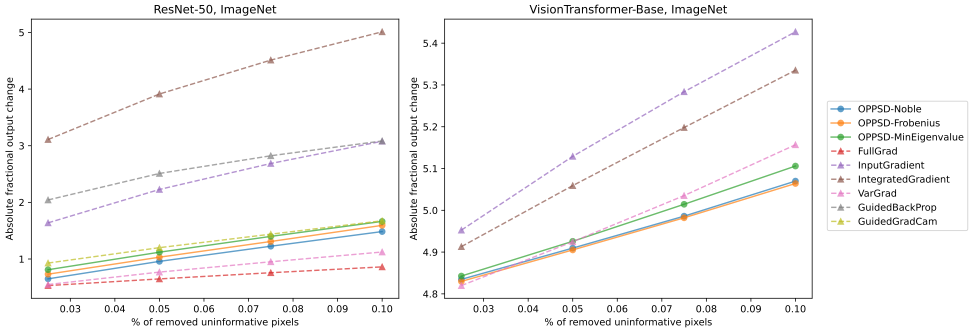


Figure 3: Pixel perturbation benchmarks with ResNet-50 (left) and VisionTransformer-Base models (right) for ImageNet validation dataset.

The pixel perturbation benchmark is the most common approach for evaluating saliency methods [4, 5, 21]. The model outputs should be affected by the information quantity of the removed pixels from the input image. Inspired by this intuition, the pixel perturbation benchmark evaluates the fractional output changes of the model regarding the percentage of removed pixels. Two types of removal strategies exist: (1) removing the top-*k*% of the informative pixels and (2) removing the top-*k*% uninformative pixels. We followed the same strategy as that in Srinivas and Fleuret [29] to remove the uninformative pixels rather than removing the informative pixels. In this strategy, the least difference in the fractional output means that the saliency method can better discriminate the uninformative pixels. We intended to evaluate the performance of our method in discriminating uninformative pixels qualitatively according to the pixel perturbation benchmark.

Figure 3 presents the pixel perturbation benchmarks that were conducted with two different model architectures on the ImageNet validation dataset [20]. Our methods are indicated by the solid lines.

The other saliency methods for comparison are indicated by the dashed lines. According to the lower the curve, the saliency method was better for correctly quantifying the uninformative pixels. For the ResNet-50 model, our methods exhibited competitive results except for FullGrad and VarGrad. For the VisionTransformer-Base model, our methods provided the most outstanding performance. Although our methods exhibited competitively but not the most outstanding performance in the ResNet-50 model experiment, they could compensate for the drawbacks of VarGrad (requiring multiple backward propagation steps) and FullGrad (not applicable to all model architectures).

4.3 ROAR Benchmark

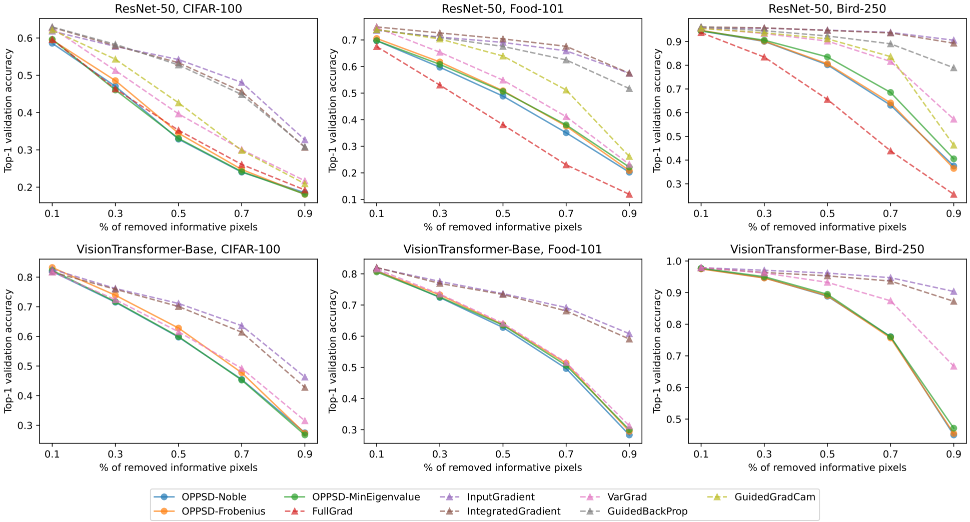


Figure 4: ROAR benchmark with ResNet-50 (top) and VisionTransformer-Base (bottom) models for three datasets: CIFAR-100 (left), Food-101 (middle), and Bird-250 (right).

ROAR [12] is a new approach for evaluating saliency methods. Unlike the pixel perturbation benchmark, ROAR retrains the model after removing the top informative pixels ranked by the saliency map. This is because the input image with partially removed pixels diverges from the training data distribution. To follow the general assumption of machine learning whereby the distributions of the training and validation data are equivalent, ROAR retrains the model from the modified training dataset and performs evaluation using the modified validation dataset. By removing the informative pixels, ROAR measures the degradation of the validation accuracy. As indicated by the lower curve, the saliency method could distinguish between the informative pixels and others better.

As illustrated in Figure 4, we conducted the ROAR benchmark with the same two models as when using the pixel perturbation benchmark on three datasets: CIFAR-100 [14], Food-101 [7], and Bird-250 [18]. Our methods are indicated by the solid lines. The other baselines are indicated by the dashed lines. Apart from the results of the Food-101 and Bird-250 datasets using the ResNet-50 model, our methods exhibited the sharpest declination in the validation accuracy curves. This empirically demonstrates that our methods provided the best performance in discriminating informative pixels in various datasets and model architectures. In addition, three corner criteria show similar performances. It firmly supports that our main idea of considering all directions of perturbation is effective on explaining the model behavior. In the next section, we will discuss this statement in more detail along with the ablation study.

4.4 Ablation Study

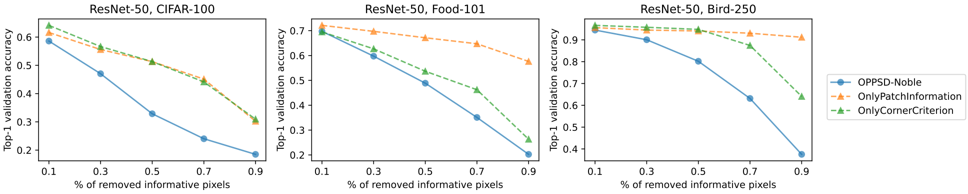


Figure 5: Ablation study on ROAR benchmark performed with ResNet50 model for three datasets: CIFAR-100 (left), Food-101 (middle), and Bird-250 (right).

As mentioned previously, the two main components of our methods are (1) collecting information from a patch centered around the target pixel and (2) the corner criterion in an eigenvalue analysis of first-order Taylor expansion. In Figure 5, we demonstrate that the two components are essential to our method by means of the ablation study.

- Only patch information (dashed-dot lines with diamond markers): without the corner criterion, averaging the input gradient over the 7×7 patch centered at the target pixel.
- Only corner criterion (dashed lines with triangular markers): without collecting patch information, using M_{noble} and restricting $k = 1$ in Equation 2.

In both cases, all experiments revealed a degradation in the ROAR benchmark compared to the original method (solid lines with circle markers). These results indicate that our methods are not solely dependent on smoothing the input gradients by spatially collecting information through the patch. It can be observed that using the corner criterion is more crucial to achieving better performance as opposed to collecting patch information. However, the combination of both components develops a synergy for a more accurate saliency map.

4.5 Computational Efficiency

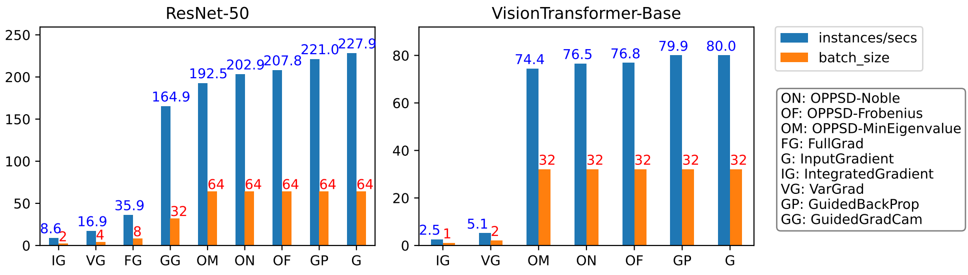


Figure 6: Computational efficiency and memory requirements.

In Figure 6, we compare the computational efficiency and memory requirements of our methods with other baselines. Our method is 5.36× faster and requires 8× less memory than FullGrad. The difference is even more remarkable compared to IntegratedGradient and VarGrad, where multiple backward propagation steps are indispensable. Our results are similar to InputGradient, which requires only a vanilla input-gradient. In other words, an additional step in our method, computation of eigenvalues, requires only small computations for most input types (RGB images).

4.6 Visual assessment

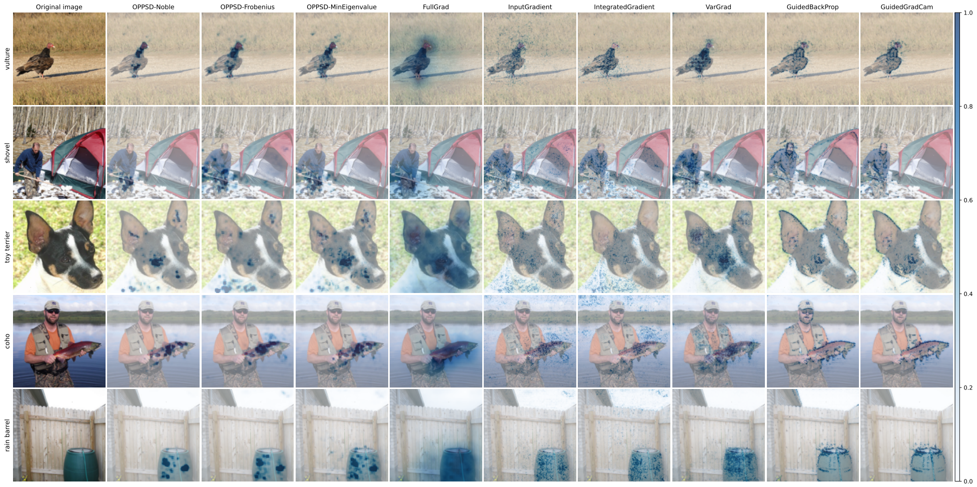


Figure 7: Visual comparisons of different saliency methods (our methods are in column 2, 3, and 4). More examples are provided in the supplement materials.

In Figure 7, we present examples of different saliency methods. We could confirm our method does not exploit edge features in images, unlike GuidedBackProp and GuidedGradCam. Our method focuses more on small areas than FullGrad but shows better visual clarity compared to InputGradient. Focusing on the small area is advantageous when the small object exists in the image (1st and 4th rows in Figure 7). If the object is large, it is helpful to find the individual descriptive parts of the object. For example, in the 3rd row in Figure 7, our method points to the dog's nose, eyes, and ears. These are the most descriptive shapes of the dog. Our three corner criteria showed similar results in the quantitative results. However, in the visual assessment, OPPSD-Noble shows the best visual clarity among them.

5 Conclusions

In this paper, we have proposed a novel perturbation-based saliency method to obtain a pixel-level saliency map. Although the saliency maps are bound to exhibit ambiguity from the definition of the explanation, perturbation-based saliency methods offer the advantage of providing a clear and intuitive definition. However, previous perturbation-based saliency methods have been difficult to apply in practice owing to the high computational costs. Our methods only require a backward propagation step of the DNN, making it affordable to obtain the pixel-level saliency map. The extensive experiments across the various datasets and two different model architectures (ResNet and VisionTransformer) reliably demonstrated that our method performs competitively under general circumstances. Furthermore, sanity checks by the model parameter randomization tests ensured that our method can correctly explain the model behavior.

References

[1] J. Adebayo, J. Gilmer, I. Goodfellow, and B. Kim. Local explanation methods for deep neural networks lack sensitivity to parameter values. *arXiv preprint arXiv:1810.03307*, 2018.

[2] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. *Advances in Neural Information Processing Systems*, 31:9505–9515, 2018.

[3] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

[4] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek. " what is relevant in a text document?": An interpretable machine learning approach. *PLOS ONE*, 12(8):e0181142, 2017.

[5] S. Bach, Alexander A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, 2015.

[6] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[7] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[8] Huiqi Deng, Na Zou, Mengnan Du, Weifu Chen, Guocan Feng, and Xia Hu. A unified taylor framework for revisiting attribution methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11462–11469, 2021.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[10] C. Guan, X. Wang, Q. Zhang, R. Chen, D. He, and X. Xie. Towards a deep and unified understanding of deep neural models in nlp. In *International Conference on Machine Learning*, pages 2454–2463, 2019.

[11] C. Harris and M. Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[12] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 9737–9748, 2019.

[13] N. Kokhlikyan, V. Miglani, M. Martin, E. Wang, B. Alsallakh, J. Reynolds, A. Melnikov, N. Kliushkina, C. Araya, S. Yan, and O. Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

[14] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[15] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[16] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[17] J. A. Noble. Finding corners. *Image and Vision Computing*, 6(2):121–128, 1988.

[18] G. Piosenka. 250 bird species. https://www.kaggle.com/gpiosenka/100-bird-species, 2020. Accessed: 2020-01-20, Version: 35.

[19] M. T. Ribeiro, S. Singh, and C. 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.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[21] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673, 2016.

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[23] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153, 2017.

[24] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, Lucas L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[25] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[26] Sahil Singla, Eric Wallace, Shi Feng, and Soheil Feizi. Understanding impacts of high-order loss approximations and features in deep learning interpretation. In *International Conference on Machine Learning*, pages 5848–5856. PMLR, 2019.

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

[28] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015. URL http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a.

[29] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network visualization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[30] M. Sundararajan, T. Ankur, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning – Volume 70*, pages 3319–3328. JMLR. org, 2017.

[31] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[32] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *5th International Conference on Learning Representations (ICLR) 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BJ5UeU9xx.