Explaining image classifiers by removing input features using generative models

Chirag Agarwal and Anh Nguyen

Auburn University, Auburn, AL 36849, USA
{chiragagarwal12,anh.ng8}@gmail.com

Abstract. Perturbation-based explanation methods often measure the contribution of an input feature to an image classifier's outputs by heuristically removing it via e.g. blurring, adding noise, or graying out, which often produce unrealistic, out-of-samples. Instead, we propose to integrate a generative inpainter into three representative attribution methods to remove an input feature. Our proposed change improved all three methods in (1) generating more plausible counterfactual samples under the true data distribution; (2) being more accurate according to three metrics: object localization, deletion, and saliency metrics; and (3) being more robust to hyperparameter changes. Our findings were consistent across both ImageNet and Places365 datasets and two different pairs of classifiers and inpainters.

1 Introduction

Explaining a classifier's outputs given a certain input is increasingly important, especially for life-critical applications [1,2]. A popular means for visually explaining an image classifier's decisions is an *attribution map* i.e. a heatmap that highlights the input pixels that are the evidence for and against the classification outputs [3]. To construct an attribution map, many methods approximate the attribution value of an input region by the classification probability change when that region is absent i.e. removed from the image. While removing an input feature to measure its attribution is a principle method (i.e. "intervention" in causal reasoning [4]), a key open question is: **How to remove?**

State-of-the-art perturbation-based attribution methods implement the absence of an input feature by replacing it with (a) mean pixels [5,6]; (b) random noise [7,8]; or (c) blurred versions of the original content [9,10]. However, these removal (i.e. perturbation) techniques often produce unrealistic, out-of-distribution images (Fig. 1b,d) on which the classifiers were not trained. Because classifiers are often easily fooled by unusual input patterns [11,12,13], we hypothesize that such examples might yield heatmaps that are (1) unreliable i.e. sensitive to hyperparameter settings [14]; and (2) not faithful [15].

To combat these two issues, we propose to harness a state-of-the-art generative inpainting model (hereafter, an inpainter) to remove pixels from an input image and fill in with content that is plausible under the true data distribution. We test our approach on three representative attribution methods of Sliding-Patch (SP) [5], LIME [6], and Meaningful-Perturbation (MP) [9] across two large-scale datasets of ImageNet [16] and Places365 [17]. For each dataset, we use a separate pair of pre-trained image classifiers

(a) Real + BB   (b) SP [5]   (c) SP-G   (d) LIME [6] (e) LIME-G   (f) MP2   (g) MP2-G

freight car 0.832   0.391   0.840   0.003   0.898   0.001   0.001
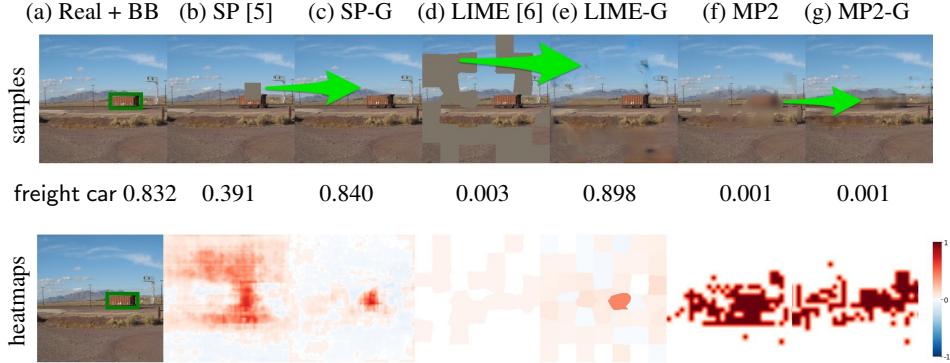

Fig. 1: Three attribution methods, SP [5], LIME [6], and MP2, often produce unrealistic, out-of-distribution perturbation samples. **Top row:** SP slides a 29×29 gray patch across the image (b). LIME grays out a set of random superpixels (d). MP2 blurs out the entire image (f). In contrast, a learned inpainter integrated into these methods produces realistic samples for the same perturbation masks, here, completing the freight car (c), completing the background (e), and removing the car from the scene (g). Note that the freight car class probability is reduced by 53% (i.e. from 0.832 to 0.391) when only a part of the car was occluded (b). However, it is reduced by ∼100% down to 0.003 when the car is still present but the background is unnaturally masked out (d). Since the inpainted samples are more realistic, the probability drops are often less (c & e) and substantial only when the object is removed completely (g). **Bottom row:** the inpainted samples yield heatmaps that, in overall, outperform the original methods on the object localization task Sec. 4.3. Here, our heatmaps (SP-G, LIME-G, and MP2-G) are less noisy and more focused on the object.

and inpainters. We chose SP, LIME, and MP because they are among the most commonly used and applicable to any classifier. Our main findings include:[1]

1. Inpainting is more effective than common techniques in removing discriminative features. That is, photos with the main object blurred or grayed out are still 3 times more recognizable by classifiers and more similar to the original photo (via MS-SSIM and LPIPS) than photos with objects removed via inpainting (Sec. 4.2).

2. Our results are the first to show that incorporating an inpainter improves perturbation-based attribution methods i.e. producing (1) more plausible perturbation samples; (2) explanations that are similarly or more accurate on three common benchmarks— object localization, deletion, and saliency metrics (Sec. 4.3); and (3) explanations that are more robust to hyperparameter changes i.e. the SAM metric [14] (Sec. 4.4);

3. We propose MP2-G (Sec. 3.5), a variant that is substantially more accurate, reliable, and having four hyperparameters fewer than the common MP [9]—a state-of-the-art approach which is the basis for many extensions [18,19,20,21,22].

To the best of our knowledge, this is the first work that shows the effectiveness of generative models in improving the accuracy and reliability of explanation methods.

---

[1] All our codes are available at https://github.com/anguyen8/generative-attribution-methods.

2   Related work

Attribution methods can be grouped into two main classes: (1) white- and (2) black-box.
White-box Given access to the network architecture and parameters, attribution maps
can be constructed analytically from (a) the gradients of the output w.r.t. the input [23],
(b) the class activation map in fully-convolutional neural networks [24], (c) both the
gradients and activations [25], or (d) the gradient times the input image [26]. However,
some gradient-based heatmaps can be too noisy to be human-interpretable [27], and
suffer from gradient saturation [28]. To combat these issues, perturbation techniques
were also utilized. That is, to make a gradient-based heatmap more robust and smooth,
a number of methods essentially average out the resultant heatmaps across a large set
of perturbed inputs that are created via (a) adding random noise to the input [9,27], (b)
blurring the input [9], or (c) linearly interpolating between the input and a reference
"baseline" image [28].
Black-box Perturbation-based methods are important for use cases when only a black-
box model is given (no network parameters). Black-box methods often remove (i.e.
perturb) an input region and take the resultant classification probability change to be
the attribution value for that region. While the idea is principle in causal reasoning, the
physical interventions—taking an object out of a scene (revealing the content behind it)
while keeping other factors unchanged—are impractical in most real-world applications.
The absence of an input region is often implemented by replacing it with (a) mean pixels
[5,6]; (b) random noise [7,8]; or (c) blurred versions of the original content [9]. However,
these removal techniques often produce unrealistic, out-of-samples (Fig. 1), which raises
huge concerns on the sensitivity and faithfulness of explanations.
An open question for existing perturbation-based attribution methods is: *Do explana-
tions become more robust and accurate if input features are removed via a strong, natural
image prior?* Here, we systematically study this question across three representative
attribution methods: two black-box methods that are perturbation-based (i.e. SP and
LIME) and one white-box method that relies on both perturbations and gradients (i.e.
MP). These representative methods also perturb different types of input features: pixels
(i.e. MP), superpixels (i.e. LIME); and square patches (i.e. SP).

The closest to our work is FIDO-CA [29], which extended MP and harnessed an
image inpainter to synthesize counterfactual samples to explain classifiers' decisions.
However, FIDO-CA [29] underperformed most baselines that do not use inpainters.
Inspired by [29], we propose a key change in optimization objectives (see details in
Sec. 4) that enabled our approach to improve upon FIDO-CA by a large margin. That
is, for the first time, we show that incorporating an inpainter improves the accuracy and
robustness of explanation methods.

3   Methods

3.1   Datasets and Networks

Classifiers   Our experiments were conducted using two separate ResNet-50 image
classifiers [30] that were pre-trained on the 1000-class ImageNet 2012 [16] and Places365

[17], respectively. The two models were officially released by the PyTorch model zoo [31] and by the authors [32], respectively.

Datasets   We chose these two datasets because they are large, natural-image sets covering a wide range of images from object-centric (i.e. ImageNet) to scenery (i.e. Places365). From the 50,000 ImageNet and 36,500 Places365 validation-set images, we randomly sampled a set of 2000 images correctly classified by their respective ResNet-50 models. We used these two sets of images in all experiments throughout the paper.

Inpainters   For each classifier, pre-trained either on ImageNet or Places365, we used a TensorFlow DeepFill-v1 model pre-trained by [33] on the same respective dataset. DeepFill-v1 takes as input a color image and a binary mask, both at resolution 256×256, and outputs an inpainted image of the same size. In this work, we also tried DeepFill-v2 [34], a free-form inpainting model, but the overall results did not change significantly. Apart from these two, to the best of our knowledge, there are no other publicly available generative inpainters for both ImageNet and Places365 datasets. The DeepFill-v1 inpainter is practically feasible for attribution algorithms as it only takes 0.2s/image on one GPU (and 1.5s/image on CPUs) for inpainting a 512×512 image.

3.2   Problem formulation

Let $s : \mathbb{R}^{D \times D \times 3} \to \mathbb{R}$ be an image classifier that maps a square, color image $\boldsymbol{x}$ of spatial dimension $D \times D$ onto a softmax probability of a target class. An attribution map $\boldsymbol{A} \in [-1, 1]^{D \times D}$ associates each input pixel $\boldsymbol{x}_i$ to a scalar $\boldsymbol{A}_i \in [-1, 1]$ which indicates how much $\boldsymbol{x}_i$ contributes for or against the prediction score $s(\boldsymbol{x})$. We describe below three methods for generating attribution maps together with our respective proposed variants (hereafter, G-methods) which harness a generative inpainter.

3.3   Sliding-Patch (SP)

SP [5] proposed to slide a gray, occlusion patch across the image and record the probability changes as attribution values in corresponding locations in the heatmap. That is, given a binary occlusion mask $m \in \{0, 1\}^{D \times D}$ (here, 1's inside the patch region and 0's otherwise) and a filler image $\boldsymbol{f} \in \mathbb{R}^{D \times D \times 3}$, a perturbed image $\bar{\boldsymbol{x}} \in \mathbb{R}^{D \times D \times 3}$ (see Fig. 1b) is given by:

$$\bar{\boldsymbol{x}} = \boldsymbol{x} \odot (1 - m) + \boldsymbol{f} \odot m \tag{1}$$

where $\odot$ denotes the Hadamard product and $\boldsymbol{f}$ is a zero image i.e. a gray image[2] before input pre-processing. For every pixel $\boldsymbol{x}_i$, one can generate a perturbation sample $\bar{\boldsymbol{x}}^i$ (i.e. by setting the patch center at $\boldsymbol{x}_i$) and compute the attribution value $\boldsymbol{A}_i = s(\boldsymbol{x}) - s(\bar{\boldsymbol{x}}^i)$. However, sliding the patch densely across the 224×224 input image is prohibitively slow. Therefore, we chose a 29×29 occlusion patch size with stride 3, which yields a smaller heatmap $\boldsymbol{A}'$ of size 66×66. We bi-linearly upsampled $\boldsymbol{A}'$ to the image size to create the full-res $\boldsymbol{A}$. See Fig. 1b for an example of SP heatmaps and perturbed images.

We implemented SP by converting a MATLAB implementation [35] into PyTorch. All of our individual experiments in this work were run on a single GTX 1080Ti GPU.

---

[2] The ImageNet mean pixel is gray (0.485, 0.456, 0.406).

SP-G Note that the stride, size, and color of a SP sliding patch are three hyperparameters that are often chosen heuristically, and varying them can change the final heatmaps radically [14]. To ameliorate the sensitivity to hyperparameter choices, we propose a variant called SP-G by only replacing the gray filler image of SP with the output image of an inpainter (described in Sec. 3.1) i.e. $\boldsymbol{f} = G(m, \boldsymbol{x})$ while keeping the rest of SP the same (Fig. 1b vs. c; top row). That is, at every location of the sliding window, SP-G queries the inpainter for content to fill in the window.

3.4 Local Interpretable Model-Agnostic Explanations (LIME)

LIME While SP occludes one square patch of the image at a time, LIME [6] occludes a random-shaped region. The algorithm first segments the input image into S non-overlapping superpixels [36]. Then, LIME generates a perturbed image $\bar{\boldsymbol{x}}$ by graying out a random set of superpixels among 2^S possible sets. That is, LIME follows Eq. 1 where the pixel-wise mask m is derived from a random superpixel mask $m' \in \{0, 1\}^S$. For each sample $\bar{\boldsymbol{x}}^i$, we measure the output score $s(\bar{\boldsymbol{x}}^i)$ and evenly distribute it among all occluded superpixels in $\bar{\boldsymbol{x}}^i$. Each superpixel's attribution is then inversely weighted by the L_2 distance $\|\boldsymbol{x} - \bar{\boldsymbol{x}}^i\|$ via an exponential kernel and then averaged out across N samples. The resultant attribution $\boldsymbol{a}_k$ of a superpixel k is finally assigned to all pixels in that group in the full-resolution heatmap $\boldsymbol{A}$.

In practice, [6] iteratively optimized for $\{\boldsymbol{a}_k\}_S$ via LASSO for 1000 steps to also maximize the number of zero attributions i.e. encouraging simpler, sparse attribution maps. We used the implementation provided by the authors of LIME [37] and their default hyperparameters of $S = 50$ and $N = 1000$.

LIME-G While avoiding the bias given by the SP square patch, LIME perturbation samples remain unrealistic. Therefore, we propose LIME-G, a variant of LIME, by only changing the gray image $\boldsymbol{f}$ to a synthesized image $G(m, \boldsymbol{x})$ as in SP-G while keeping the rest of LIME unchanged.

3.5 Meaningful Perturbation (MP)

MP As SP and LIME gray out patches and superpixels in the input image, they generate unrealistic counterfactual samples and produce coarse heatmaps. To combat these issues, Fong et al. [9] proposed the MP algorithm i.e. learning a minimal, continuous mask $m \in [0, 1]^{D \times D}$ that blurs out the input image in a way that would minimize the target-class probability. That is, MP attempts to solve the following optimization problem:

$$m^* = \arg\min_m \lambda\|m\|_1 + s(\bar{\boldsymbol{x}}) \tag{2}$$

where $\bar{\boldsymbol{x}}$ is given by Eq. 1 but with $\boldsymbol{f} = B_\sigma(\boldsymbol{x})$ i.e. the input image blurred by a Gaussian kernel $B_\sigma(.)$ of radius $\sigma = 10$. Note that, in MP, the attribution map $\boldsymbol{A}$ is also the learned mask m. However, solving Eq. 2 directly often yields heatmaps that are noisy and sensitive to hyperparameter changes [14]. Therefore, MP only learned a small 28×28 mask and upsampled it to the image size in every optimization step. In addition,

they also encouraged the mask to be smooth and robust to input changes by changing the objective function to the following:

$$m^* = \arg\min_m \lambda_1 \|m\|_1 + \lambda_2 TV(m) + \mathbb{E}_{\tau \sim \mathcal{U}(0,4)}\big[s(\Phi(\bar{x}, \tau))\big] \tag{3}$$

where $TV(m) = \sum_i \|\nabla m_i\|_3^3$ i.e. a total-variation norm that acts as a smoothness prior over the mask. The third term is the expectation over a batch of randomly jittered versions of the blurred image $\bar{x}$. That is, $\Phi(.)$ is the jitter operator that translates an image $\bar{x}$ vertically or horizontally by τ pixels where τ is drawn from a discrete uniform distribution $\mathcal{U}(0,4)$. We randomly initialized the mask from a continuous uniform distribution $\mathcal{U}(0,1)$ and minimize the objective function in Eq. 3 via gradient descent for 300 steps. Our MP implementation was in PyTorch and followed all the hyperparameters as described in [9].

MP2   In the original formulation, MP is highly sensitive to changes in some of its hyperparameters [14]. In our preliminary experiments (data not shown), we found that integrating an inpainter into the existing unstable MP optimization did not yield more accurate heatmaps. In addition, the L_1 and TV terms (Eq. 3) introduce strong biases that impede the contribution of the content generated by inpainters.

Therefore, we propose MP2, a more reliable and accurate variant by eliminating four hyperparameters from MP: the L_1 norm, TV norm, the jitter operator and the stopping criterion of 300 optimization steps (Sec. 4.3). That is, we still find a minimal mask (Eq. 2) but by initializing it with all zeros and growing the number of 1's (i.e. the blurred region) gradually. Following JSMA [38], in every iteration, we add 1's to two pixels that have the highest gradient norms. We stop the mask optimization when the classification probability reaches random chance, i.e. 0.001 for ImageNet and 0.003 for Places365. As MP, we use the same Gaussian blur radius of 10 and the mask size of 28×28.

MP2-G   We integrate an inpainter G to MP2 by only changing the filler image $f = B_\sigma(x)$ that is used in Eq. 1 to an inpainted image i.e. $f = G(m_b, x)$ where $m_b \in \{0, 1\}^{D \times D}$ is the binary mask learned via MP2 optimization.

4   Experiments and Results

4.1   Inpainter failed to synthesize backgrounds given only foreground objects

Chang et al. [29] proposed to find a minimal set of input pixels that would keep the classification outputs unchanged even when the other pixels in the image are removed (i.e. the "preservation" objective [9]) via an inpainter. Their method, FIDO-CA, uses the same DeepFill-v1 inpainter as in our work; however, their "preservation" objective encourages the inpainter to predict the missing background pixels conditioned on the remaining foreground object—a task that DeepFill-v1 was *not* trained to do and thus produced unrealistic samples as in [29]. In contrast, our MP2-G method harnesses the dual "deletion" objective i.e. finding the smallest set of input pixels which when inpainted would minimize the target-class probability—which intuitively asks the inpainter to replace the main object with some content that is consistent with the background i.e. the training objective of DeepFill-v1.
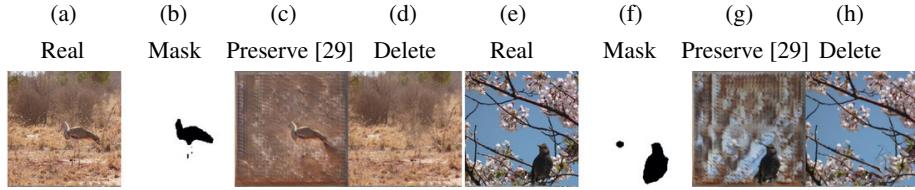
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Real	Mask	Preserve [29]	Delete	Real	Mask	Preserve [29]	Delete

Fig. 2: Using the DeepFill-v1 inpainter to fill in the background region (i.e. "preservation" task [29]) yields unrealistic images that contain features unnaturally pasted from the object (c, g). This key difference between the "deletion" (d, h) and "preservation" (c, g) objectives is further reflected in the evaluation results of MP2-G and FIDO-CA [29] where the attribution maps generated using the latter consistently underperforms than MP2-G (Sec. 4.3). See Fig. S3 for more examples of the images.

(a) Real	(b) Blur	(c) Gray	(d) Inpaint	(e) Noise

bustard 0.996	0.020	0.050	0.001	0.001

Fig. 3: The results of filling the object mask in a real image (a) via four different filling methods. The shape of the bird is still visible even after blurring (b), graying out (c) or adding random noise (e) to the bird region. The inpainter removes the bird and fills in with some realistic background content (d).

To compare these two objectives, we randomly chose 50 validation-set images from 52 ImageNet bird classes and computed their segmentation masks via a pre-trained DeepLab model [39]. We found that using the DeepFill-v1 to inpaint the foreground region (i.e. our "deletion" task) yields realistic samples where the object is removed. In contrast, using the inpainter to fill in the missing background area [29] yields unrealistic images whose backgrounds contain features (e.g. bird feathers or beaks) unnaturally pasted from the object (Fig. 2). This result motivated us to integrate DeepFill-v1 into MP2 but with the "deletion" objective.

4.2 Inpainter is effective in removing discriminative features

While removing objects from an image via DeepFill-v1, qualitatively, yields realistic samples, here, we quantitatively test how effective this strategy is in removing target-class discriminative features in comparison with three existing filling methods: (1) zero pixels; (2) random noise; or (3) blurred versions of the original content. Using the same procedure as described in Sec. 4.1, we randomly sampled 1000 bird images and segmented out the bird in each image. We filled in the object mask in each image via all four methods (Fig. 3) and compared the results (Table 1). Surprisingly, the blurred and grayed-out images are still correctly classified at 26.4% and 13.3% (Table 1), respectively, by a pre-trained Inception-v3 classifier [40], i.e., these perturbed images still contain discriminative features relevant to the target class. In contrast, only 8.9% of the inpainted

images were correctly classified suggesting that the inpainter removes the discriminative features more effectively. After the main subject (here, birds) are removed from an image, one would expect the modified image to be perceptually different from the original image (where the bird exists).

Table 1: Evaluation of four different filling methods on 1000 random bird images. The Inception-v3 accuracy scores suggest that inpainting the object mask (d) removes substantially more discriminative features relevant to the removed object compared to blurring (b) or graying out (c). Perceptually, the inpainted images are also more dissimilar to the corresponding real images according two similarity metrics: MS-SSIM (lower is better) and LPIPS (higher is better).

Metrics	Filling methods				
	(a) Real	(b) Blur	(c) Gray	(d) Inpaint	(e) Noise
Inception Acc.(%)	92.30	26.40	13.30	8.90	4.40
MS-SSIM	1.000	0.941	0.731	0.707	0.692
LPIPS	0.000	2.423	3.186	3.208	3.639

Here, we evaluate how each of the four in-filled images $\bar{x}$ (where the bird has been removed) is perceptually dissimilar to the original image x by measuring the MS-SSIM and LPIPS [41] scores between every pair $(x, \bar{x})$. Across both metrics, the inpainted images are consistently more dissimilar from the real images compared to the blurred and grayed-out images. Note that in all three quantitative metrics, the inpainted images are the closest to the noise-filled images (Table 1d–e) despite being substantially more realistic (Fig. 3). Furthermore, the problem with using blurring as a perturbation operation is explicitly seen in cases where attribution maps covers the entire image. This is because for some inputs even blurring out the entire image does not result in a significant probability (Fig. 4). Across the set of 2000 images, the average confidence score on blurring the entire image was 0.3198.

0.047   0.046   0.871   0.022   0.025   0.017   0.174   0.929   0.015
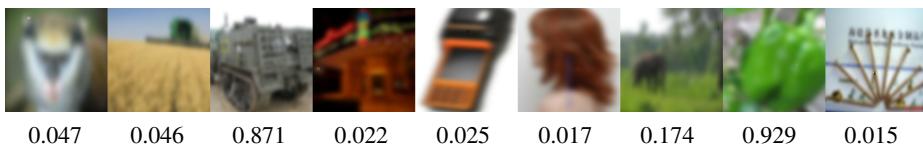
Fig. 4: The target class probability of images do not drop to random guess, i.e. 0.001 for ImageNet, even after perturbing the entire image with a Gaussian blur radius of $\sigma =10$.

4.3   Are explanations by G-methods more accurate?

While there are currently no established ground-truth datasets to evaluate the correctness of an attribution map, prior research often assessed correctness via three common proxy metrics: (1) the object localization task [24]; (2) the deletion task [42]; (3) the saliency metric [7]. Here, we ran 8 algorithms on the ImageNet and Places365 datasets using the

Table 2: Localization errors (lower is better) for all attribution methods on ImageNet. Naively taking the whole image as a bounding box yields an error of 38.56% (baseline). MP2-G outperformed all methods including MP, MP2 and a related FIDO-CA [29].

Baseline	SP [5]	SP-G	LIME [6]	LIME-G	MP [9]	MP2	MP2-G	FIDO-CA [29]
39.7%	41.9%	**38.95%**	28.05%	**26.55%**	29.35%	24.4%	**24.03%**	27.9%

default hyperparameters (Sec. 3). The heatmaps are then upsampled to the full image resolution for evaluation on all three measures above.

Object localization Zhou et al. [24] proposed to evaluate heatmaps by localizing objects in the ImageNet images, which often contain a single object of a known class. We followed the localization procedure in [9] for the ImageNet dataset. That is, for each algorithm, we derived multiple bounding boxes per heatmap by thresholding it at different values of $t = \alpha\mu_{max}$, where μ_{max} is the maximum intensity in the heatmap and $\alpha \in [0 : 0.05 : 0.95]$. For each α, we computed the Intersection over Union (IoU) score between a derived bounding box and the ImageNet ground-truth. The object localization error was calculated by thresholding each IoU score at 0.5 and averaging them across the number of images. For each method, we chose the best α^* that yielded the lowest error on a held-out set of 1000 ImageNet images (Table 2). **We found that our generative version of the attribution algorithms outperformed their respective counterparts and MP2-G outperformed FIDO-CA**[3] **(Table 2).** Among the 8 methods, MP2-G obtained the lowest error of 24.03%. Qualitatively, MP2-G generates attribution maps that are more localized to the objects in the image (Fig. 5).

Table 3: **Deletion metric** (lower is better): SP-G, LIME-G, and MP2-G outperformed their counterparts on both ImageNet and Places365 datasets. G-methods also outperformed a baseline (here, random attribution maps).

Dataset	Baseline	SP[5]	SP-G	LIME[6]	LIME-G	MP[9]	MP2	MP2-G	FIDO-CA [29]
ImageNet	0.2083	0.1996	**0.1769**	0.1355	**0.1171**	0.1654	0.1530	**0.1311**	0.1638
Places365	0.2151	0.2560	**0.1944**	0.1919	**0.1582**	0.2014	0.1980	**0.1871**	0.1987

Deletion metric Intuitively, if the attributions in an explanation correctly reflect the importance of input pixels, removing the input pixels of highest attributions should cause a substantial probability drop. The deletion metric [42] measures the area under the curve of the target-class probability as we gradually zero out input pixels of the highest attributions in descending order. The deletion scores are widely used to compare attribution methods [43,18,44,45] i.e., lower deletion scores are considered more accurate. Here, we evaluated all 8 methods via the code released by [42] where the authors knocked out 224×8 pixels at a time. Similar to object localization results, we observed a consistent trend: **Across both ImageNet and Places365, our G-methods outperformed their counterparts while MP2-G outperformed all algorithms** (Table 3).

---

[3] We produced FIDO-CA results using the code provided by the authors [29]. See Sec. A1 for more details.

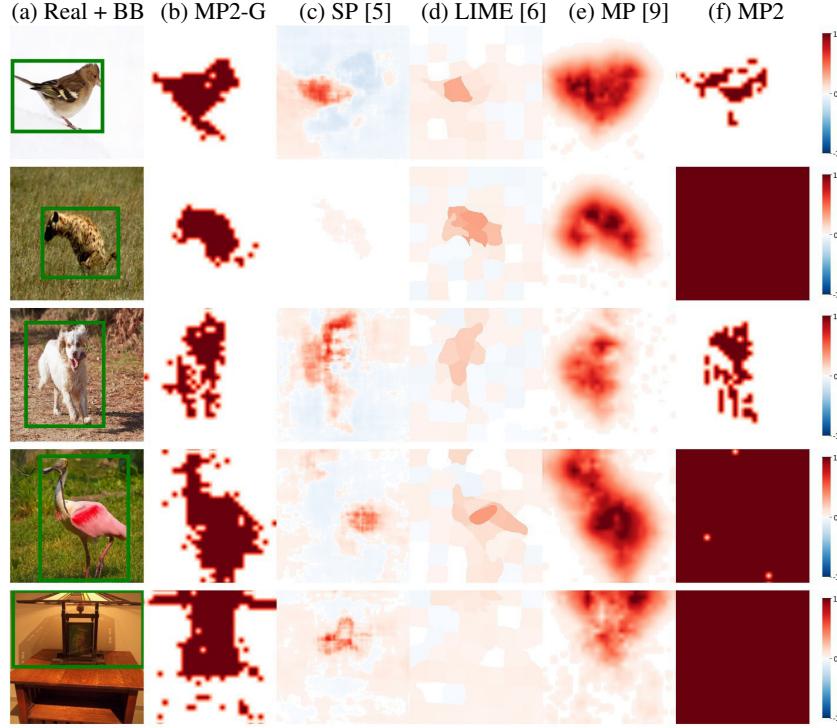(a) Real + BB   (b) MP2-G   (c) SP [5]   (d) LIME [6]   (e) MP [9]   (f) MP2

Fig. 5: MP2-G results in attribution maps that localize the objects accurately compared to other perturbation-based methods. From left to right, in each row, we show a real ImageNet image with its ground-truth bounding box (BB) (a), attribution maps from the proposed MP2-G (b) and other existing methods (c–f). Images are randomly chosen. For qualitative evaluation, Figs. S4-S5 show a set of heatmaps and their derived BB's.

Saliency metric Dabkowski et al. [7] proposed that if an explanation is accurate then the most salient patch in an image (derived from the attribution map) should have a high prediction score. That is, we took the smallest rectangular patch derived from thresholding the attribution map using an α^* which yielded the least salient metric score on a held-out dataset of 1000 images (similar to the object localization task). The saliency metric is then defined as $\log\big(\max(a, 0.05)\big) - \log(s(\boldsymbol{x}_p))$ where a is the ratio of the patch size over the image size and $s(\boldsymbol{x}_p)$ is the classification probability for the patch $\boldsymbol{x}_p$ upsampled to the full image size. A lower saliency score indicates a more accurate explanation. **On both ImageNet and Places365, SP-G and MP2-G obtained lower scores than their counterparts while LIME-G was on par with LIME (Table 4). MP2-G outperformed all its baselines, i.e. MP, MP2, and FIDO-CA.** We hypothesize that the difference between LIME vs. LIME-G is small because they operate at the superpixel level and most salient *pixels* might fall in common *superpixels* across their respective explanations. Refer to Fig. S6 for the localization error and saliency metric scores for different α's on the held-out set of 1000 images.

Table 4: **Saliency metric** (lower is better): On both ImageNet and Places365, while LIME and LIME-G performed on-par, SP-G and MP2-G consistently outperformed their counterparts (MP2-G outperformed its baselines: FIDO-CA, MP and MP2). The baseline was calculated using a random attribution map.
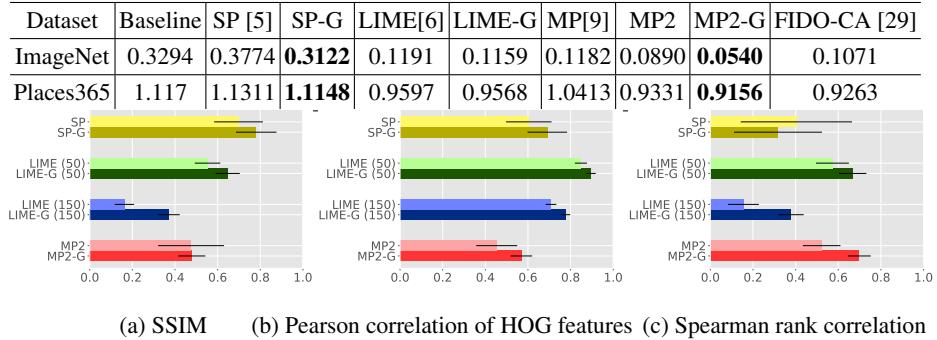
Dataset	Baseline	SP [5]	SP-G	LIME[6]	LIME-G	MP[9]	MP2	MP2-G	FIDO-CA [29]
ImageNet	0.3294	0.3774	**0.3122**	0.1191	0.1159	0.1182	0.0890	**0.0540**	0.1071
Places365	1.117	1.1311	**1.1148**	0.9597	0.9568	1.0413	0.9331	**0.9156**	0.9263

(a) SSIM   (b) Pearson correlation of HOG features (c) Spearman rank correlation

Fig. 6: Error plots for SSIM (a), Pearson correlation of HOG features (b), and Spearman rank correlation (c) scores obtained from 1000 random ImageNet images (higher is better). G-methods are more robust than their counterparts (dark vs light bars). LIME-G, in particular, is robust than LIME on both low and high resolutions i.e. $S \in \{50, 150\}$ (green and blue bars). The same trends were also observed on the Places365 dataset (Fig. S2). The exact numbers are reported in Table S1.

4.4   Are G-methods more robust to hyperparameter changes?

Machine learning methods are highly sensitive to hyperparameters, contributing to the reproducibility crisis [46]. Similarly, perturbation-based attribution methods were recently found to be highly sensitive to common hyperparameters [14]. Such sensitivity poses a huge challenge in (1) evaluating the explanations; and (2) building trust with end users [1]. Our hypothesis is that heuristically-perturbed samples are often far from the true data distribution and thus contribute to the hyperparameter sensitivity of heatmaps. Here, we test whether our generative methods are more robust to hyperparameter changes than their original counterparts.

Similarity metrics and Image sets   Following [15,14], we used three metrics from scikit-image [47] to measure the similarity of heatmaps: the Structural Similarity Index (SSIM), the Pearson correlation of the histograms of oriented gradients (HOGs), and the Spearman rank correlation. We upsampled all heatmaps to the full image size before feeding them into the similarity metrics. We performed the test on a set of 1000 random images from both ImageNet and Places365.

SP sensitivity across patch sizes   It remains a question how to choose the patch size in the SP algorithm because changing it can change the explanation radically [48]. Therefore, we compare the sensitivity of SP and SP-G when sweeping across 5 patch sizes $p \times p$ with stride 3 where $p \in \{5, 17, 29, 41, 53\}$. We chose this set to cover the common sizes that have been used in the literature. For each input image, we obtained $k = 5$ heatmaps (i.e. each corresponds to a patch size) and then measured the similarity among all $k(k-1)/2 = 10$ possible pairs.
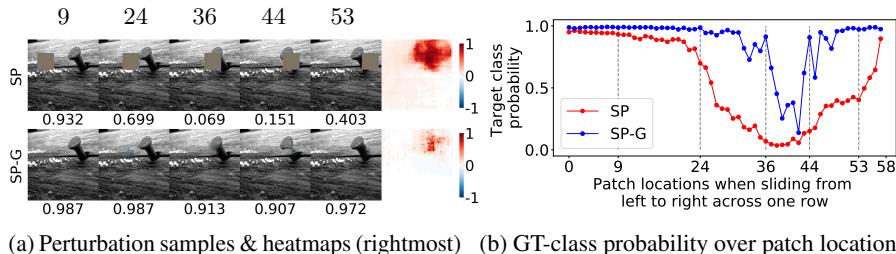
LIME sensitivity across random batches of samples   LIME randomly samples N perturbed images $\{\bar{x}^i\}_N$ and uses them to fit a heatmap. Therefore, we compared the sensitivity of LIME and LIME-G across 5 random batches of $N = 500$ perturbation samples. That is, for each input image among the 1000, we generated $k = 5$ heatmaps and computed the similarity among all 10 possible pairs. We ran this experiment for a small and a large heatmap resolution i.e. two numbers of superpixels $S \in \{50, 150\}$ while keeping all other hyperparameters constant.

MP2 sensitivity across mask sizes   Because optimizing a mask at a high resolution is prohibitively slow, Fong et al. [9] used an MP mask of size 28×28 and upsampled it to the image size when applying the blur operator on the input image. Therefore, the mask size is a hyperparameter of MP2 and MP2-G. Here, we compare the sensitivity by sweeping across the three mask sizes where $D \in \{28, 56, 112\}$. We re-ran each algorithm three times on each input image to yield three heatmaps and computed the average pairwise similarity scores from all possible pairs of heatmaps.

Results First, we found that all 6 algorithms produce inconsistent explanations across the controlled hyperparameters (Fig. 6; all scores are below 1). That is, LIME heatmaps can change as one simply changes the random seed! However, LIME-G is consistently more robust than LIME across all metrics and superpixel settings (Figs. 6 & S17). Across the patch sizes, SP-G is also consistently more robust than SP (Fig. 6a–b; light vs. dark yellow). SP-G and SP performed on par with high standard deviations under the Spearman rank correlation (Fig. 6c). Across the optimization mask size, MP2-G is consistently more robust than MP2 (Fig. 6; light vs. dark red).

5   The inner-workings of generative attribution methods

Here, we further explain why our G-methods are both more (1) accurate in localizing objects (Sec. 4.3) and (2) robust to hyperparameter changes (Sec. 4.4).

(a) Perturbation samples & heatmaps (rightmost)   (b) GT-class probability over patch locations

Fig. 7: We ran SP and SP-G using a 53×53 patch on a nail class image. Here are the perturbation samples from both methods when the patch is slid horizontally across a row at 5 locations $\{9, 24, 36, 44, 53\}$ (a); and their respective target-class probabilities (b). SP-G samples are more realistic than SP and its heatmap localizes the object accurately (a). That is, the probabilities for SP-G samples are more stable and only substantially drop when the patch covers the object (blue vs. red). See Fig. S7 for more examples.
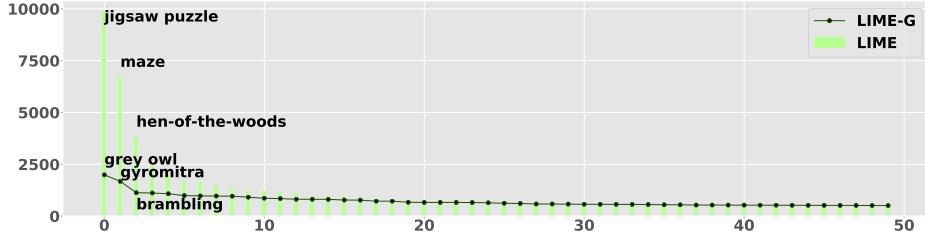
Fig. 8: We ran LIME and LIME-G on 200 images, each run has 500 intermediate perturbation samples. Here, for LIME (light green) and LIME-G (dark green) samples, we show a histogram of the top-1 predicted class labels for all 200 runs $\times 500$ samples = 100,000 images. LIME perturbed samples are highly biased towards few jigsaw puzzle, maze classes (left panel), which is somewhat intuitive given the gray-masked images (see Figs. S8–S13). In contrast, the histogram of LIME-G samples are almost uniform. **x-axis:** For visualization purposes, we sorted the top-1 labels and showed only first 50 labels. See Fig. S16 for an expanded version of the figure.

5.1 More accurate object localization: A case study of SP-G

We found that as the gray patch of SP is slid from left to right across the input image (Fig. 7a; top), the target-class probability gradually decreases and approaches 0 when the patch occludes most of the object (Fig. 7b; red line). Notably, the probability even drops when the patch is far outside the object region (Fig. 7b; red line within $[0, 24]$) due to SP's unrealistic grayish samples. Hence, the probability distributions by SP often yield a large blob of high attributions around the object in the heatmap (Fig. 7a; top-right). In contrast, the inpainted samples of SP-G often keep the probability variance low except when the patch overlaps with the object (Fig. 7b; blue vs. red), yielding heatmaps that are more localized towards the object (Fig. 7a; bottom). Across 1000 random ImageNet images, we found that the average probability change when the SP 53×53 patch is outside the object bounding box is $\sim 2.1\times$ higher than that of SP-G (i.e. 0.09 vs. 0.04). In sum, our observations here are consistent with the findings that G-methods obtained lower localization errors than the original counterparts.

5.2 More robust heatmaps: A case study of LIME-G

Here, we provide insights for why LIME-G produced heatmaps that are more consistent than LIME across 5 random batches of samples. We observed that the top-1 predicted labels of $\sim 20.5\%$ of the LIME grayish perturbation samples (e.g. Fig. 9a) were from only three classes { jigsaw puzzle, maze, hen-of-the-wood } whereas the same top-1 label distribution for LIME-G samples was almost uniform (see Fig. 8 for more details). Due to their similar grayish, puzzle-like patterns, many LIME samples across images from different classes (e.g. dogs or nail) are still classified into the same label! Relatedly, we observed that a LIME perturbation sample is often given a near-zero probability score *regardless of what input feature is being masked out* (Fig. 9a). Therefore, when fitted to N samples, where N is often too small w.r.t. the total 2^S possible samples, the heatmap appears random and changes upon a new set of random masks (Fig. 9b).

(a) 5 LIME perturbation samples

(b) 5 LIME heatmaps using five random seeds

(c) 5 LIME-G perturbation samples

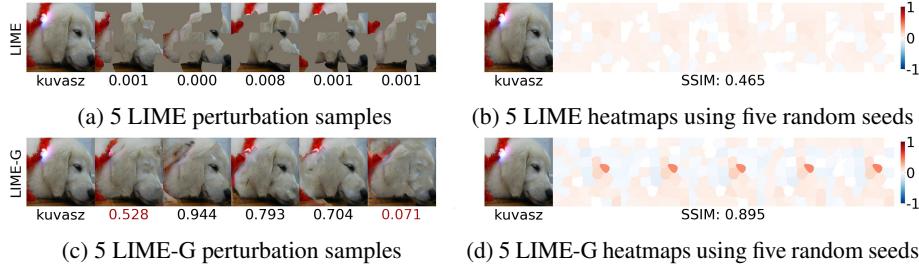(d) 5 LIME-G heatmaps using five random seeds

Fig. 9: In Sec. 4.4, we compared the robustness of LIME vs. LIME-G heatmaps when running using 5 different random seeds. This is an example where LIME-G heatmaps are more consistent than LIME's (d vs. b). While LIME grayish samples (a) are given near-zero probabilities, LIME-G samples (here, inpainted using the same masks as those in the top row) are often given high probabilities except when the kuvasz dog's eye is removed (c). LIME-G consistently assign attributions to the dog's eye (d) while LIME heatmaps appear random (b). The top-1 predicted labels for 4 out of 5 LIME samples (a) here are paper towel.

In contrast, for LIME-G samples, the probabilities consistently drop when some discriminative features (e.g. the kuvasz dog's eye in Fig. 9c) are removed. This phenomenon yields heatmaps that are more consistently localized around the same input features across different random seeds (Fig. 9d). Our explanation also aligns with the finding that when the number of superpixels S increases from 50 to 150 (while the sample size remains at $N = 500$), the sensitivity gap between LIME vs. LIME-G increases by ~ 3 times (Fig. 6a; gap between green bars vs. gap betwen blue bars). See Figs. S8–S15 for qualitative examples of when LIME-G is more robust than LIME and vice-versa. Quantitatively, we found that the image distribution where LIME-G showed superior robustness over LIME *across all three similarity metrics* mostly contains images of scenes, close-up or tiny objects. In contrast, LIME is more robust than LIME-G on images of mostly birds and medium-sized objects (See Sec. A2 for more details).

6   Discussion and Conclusion

MP2-G outperforming FIDO-CA consistently on all accuracy metrics confirms that the "deletion" objective is more appropriate for MP2 when incorporating generative inpainters. Additionally, discretizing and removing the hyperparameters of the original MP formulation aid in generating attribution maps that achieve better results across localization error, deletion, and saliency metric scores.

Integrating a state-of-the-art inpainter into three representative attribution methods consistently yielded explanations that are (1) more accurate based on three metrics; (2) more robust to hyperparameter changes; and (3) based on more plausible counterfactuals. Our results suggest that harnessing generative models to generate synthetic interventions (here, removal of input features) is a promising direction for future causal explanation methods.

Acknowledgments

References

1. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608 (2017)
2. Gunning, D., Aha, D.W.: Darpa's explainable artificial intelligence program. AI Magazine **40** (2019) 44–58
3. Montavon, G., Samek, W., Müller, K.R.: Methods for interpreting and understanding deep neural networks. Digital Signal Processing **73** (2018) 1–15
4. Hagmayer, Y., Sloman, S.A., Lagnado, D.A., Waldmann, M.R.: Causal reasoning through intervention. Causal learning: Psychology, philosophy, and computation (2007) 86–100
5. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision, Springer (2014) 818–833
6. Ribeiro, M.T., Singh, S., Guestrin, C.: 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, ACM (2016) 1135–1144
7. Dabkowski, P., Gal, Y.: Real time image saliency for black box classifiers. In: Advances in Neural Information Processing Systems. (2017) 6967–6976
8. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems. (2017) 4765–4774
9. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 3429–3437
10. Fong, R., Patrick, M., Vedaldi, A.: Understanding deep networks via extremal perturbations and smooth masks. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 2950–2958
11. Alcorn, M.A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.S., Nguyen, A.: Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 4845–4854
12. Nguyen, A., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 427–436
13. Agarwal, C., Nguyen, A., Schonfeld, D.: Improving robustness to adversarial examples by encouraging discriminative features. In: 2019 IEEE International Conference on Image Processing (ICIP), IEEE (2019) 3801–3505
14. Bansal, N., Agarwal, C., Nguyen, A.: Sam: The sensitivity of attribution methods to hyper-parameters. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2020) 8673–8683
15. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. In: Advances in Neural Information Processing Systems. (2018) 9505–9515
16. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115** (2015) 211–252
17. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE transactions on pattern analysis and machine intelligence **40** (2017) 1452–1464
18. Wagner, J., Kohler, J.M., Gindele, T., Hetzel, L., Wiedemer, J.T., Behnke, S.: Interpretable and fine-grained visual explanations for convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 9097–9107
19. Qi, Z., Khorram, S., Li, F.: Visualizing deep networks by optimizing with integrated gradients. arXiv preprint arXiv:1905.00954 (2019)

20. Carletti, M., Godi, M., Aghaei, M., Cristani, M.: Understanding deep architectures by visual summaries. arXiv preprint arXiv:1801.09103 (2018)
21. Wang, Y., Hu, X., Su, H.: Learning attributions grounded in existing facts for robust visual explanation. XAI (2018) 178
22. Uzunova, H., Ehrhardt, J., Kepp, T., Handels, H.: Interpretable explanations of black box classifiers applied on medical images by meaningful perturbations using variational autoencoders. In: Medical Imaging 2019: Image Processing. Volume 10949., International Society for Optics and Photonics (2019) 1094911
23. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034 (2013)
24. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 2921–2929
25. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 618–626
26. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In Precup, D., Teh, Y.W., eds.: Proceedings of the 34th International Conference on Machine Learning. Volume 70 of Proceedings of Machine Learning Research., International Convention Centre, Sydney, Australia, PMLR (2017) 3145–3153
27. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: Smoothgrad: removing noise by adding noise. arXiv preprint arXiv:1706.03825 (2017)
28. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org (2017) 3319–3328
29. Chang, C.H., Creager, E., Goldenberg, A., Duvenaud, D.: Explaining image classifiers by counterfactual generation. In: International Conference on Learning Representations. (2019)
30. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
31. PyTorch: torchvision.models — pytorch master documentation.   `https://pytorch.org/docs/stable/torchvision/models.html` (2019) (Accessed on 09/21/2019).
32. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Csailvision/places365: The places365-cnns for scene classification. `https://github.com/CSAILVision/places365` (2019) (Accessed on 09/21/2019).
33. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 5505–5514
34. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. arXiv preprint arXiv:1806.03589 (2018)
35. MathWorks: Network   visualization   based   on   occlusion   sensitivity.   `https://blogs.mathworks.com/deep-learning/2017/12/15/network-visualization-based-on-occlusion-sensitivity/`   (2019) (Accessed on 09/17/2019).
36. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE transactions on pattern analysis and machine intelligence **34** (2012) 2274–2282
37. Ribeiro, M.: Lime: Explaining the predictions of any machine learning classifier. `https://github.com/marcotcr/lime/` (2019) (Accessed on 09/17/2019).

38. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: 2016 IEEE European symposium on security and privacy (EuroS&P), IEEE (2016) 372–387

39. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence **40** (2017) 834–848

40. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 2818–2826

41. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 586–595

42. Petsiuk, V., Das, A., Saenko, K.: Rise: Randomized input sampling for explanation of black-box models. In: Proceedings of the British Machine Vision Conference (BMVC). (2018)

43. Arras, L., Horn, F., Montavon, G., Müller, K.R., Samek, W.: " what is relevant in a text document?": An interpretable machine learning approach. PloS one **12** (2017) e0181142

44. Hooker, S., Erhan, D., Kindermans, P.J., Kim, B.: A benchmark for interpretability methods in deep neural networks. In: Advances in Neural Information Processing Systems. (2019) 9734–9745

45. Samek, W., Binder, A., Montavon, G., Lapuschkin, S., Müller, K.R.: Evaluating the visualization of what a deep neural network has learned. IEEE transactions on neural networks and learning systems **28** (2016) 2660–2673

46. Hutson, M.: Artificial intelligence faces reproducibility crisis. Science **359** (2018) 725–726

47. van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T., the scikit-image contributors: scikit-image: image processing in Python. PeerJ **2** (2014) e453

48. Zintgraf, L.M., Cohen, T.S., Adel, T., Welling, M.: Visualizing deep neural network decisions: Prediction difference analysis. arXiv preprint arXiv:1702.04595 (2017)

A1   Implementation details of FIDO-CA

For all the results of FIDO-CA, we followed the implementation details in the code released on Github `https://github.com/zzzace2000/FIDO-saliency` by the authors [29]. FIDO-CA was ran using the "preservation" objective in conjunction with the DeepFill-v1 [33] inpainter that we also harnessed in this paper. For the optimization, we used Adam optimizer with a learning rate of 0.05 and a regularization coefficient of 0.001. A coarse 56×56 mask was optimized using a ResNet-50 classifier for the ImageNet-S and Places365-S datasets respectively. The mask was finally upsampled to the full image size, i.e., 224×224, using bilinear interpolation.

A2   LIME-G is more robust than LIME on images of scenes, close-up and tiny objects

We have shown that LIME-G is more robust than LIME consistently on all 3 different similarity metrics (see Sec. 4.4 in the main text). Here, we aim to understand the image distributions where LIME-G was more robust than LIME and vice versa.

   For each of the three metrics, we computed a set of top-100 score differences between LIME-G vs. LIME. Interestingly, we found the intersection of the three sets contains images of mostly scenes, close-up or tiny objects (see Fig. S1). In contrast, the common set of images where LIME is more robust than LIME-G contains mostly birds and medium-sized objects. These image distributions intuitively align with the domains where DeepFill-v1 is capable of inpainting and suggest that the performance of G-methods can be improved further with class-conditional inpainters.

Table S1: The results in this table are the number forms of the ImageNet sensitivity results in Fig. 6. G-methods are more robust to hyperparameters across different sensitivity metrics.

Method	Similarity Metrics		
	SSIM	Pearson correlation of HOGs	Spearman
SP	0.698±0.114	0.604±0.106	0.404±0.261
SP-G	**0.781±0.095**	**0.691±0.093**	0.317±0.206
LIME (50)	0.553±0.060	0.848±0.028	0.573±0.077
LIME-G (50)	**0.647±0.057**	**0.896±0.022**	**0.667±0.065**
LIME (150)	0.163±0.045	0.708±0.025	0.155±0.072
LIME-G (150)	**0.371±0.051**	**0.776±0.022**	**0.379±0.059**
MP2	0.476±0.155	0.453±0.096	0.522±0.088
MP-G	**0.479±0.064**	**0.569±0.051**	**0.698±0.054**

Table S2: The results in this table are the number forms of the Places365 sensitivity results in Fig. S2. The results follow the same trend as the ImageNet dataset.

Method	Similarity Metrics		
	SSIM	Pearson correlation of HOGs	Spearman
SP	0.577±0.177	0.674±0.073	0.452±0.288
SP-G	**0.720±0.122**	**0.755±0.056**	**0.332±0.208**
LIME (50)	0.392±0.074	0.802±0.036	0.594±0.078
LIME-G (50)	**0.498±0.076**	**0.865±0.027**	**0.722±0.058**
LIME (150)	0.118±0.046	0.701±0.026	0.201±0.071
LIME-G (150)	**0.312±0.061**	**0.780±0.022**	**0.511±0.051**
MP2	0.466±0.113	0.409±0.141	0.483±0.140
MP2-G	**0.494±0.053**	**0.505±0.060**	**0.618±0.057**

(a) SSIM   (b) Pearson correlation of HOG features (c) Spearman rank correlation

Fig. S2: Bar plots comparing the robustness (higher is better) of G-methods and their counterparts when changing hyperparameters (described in Sec. 4.4) under three different similarity metrics: SSIM (a), Pearson correlation of HOG features (b), and Spearman rank correlation (c). Each bar shows the mean and standard deviation similarity score across 1000 pairs of heatmaps, each produced for one random **Places365** image. G-methods are consistently more robust than their counterparts across all metrics. The exact numbers are reported in Table S2.

Images where LIME-G outperformed LIME across all three sensitivity metrics

Images where LIME-G underperformed LIME across all three sensitivity metrics

Fig. S1: Common images across all three metrics where LIME-G is consistently more robust than LIME (top) and vice versa (bottom). Interestingly, we found the intersection of the three sets contains images of mostly scenes, close-up or tiny objects (top). In contrast, the common set of images where LIME is more robust than LIME-G contains mostly birds and medium-sized objects (bottom).

(a) Real (b) Mask (c) Preserve (d) Delete (e) Real (f) Mask (g) Preserve (h) Delete

Fig. S3: Inpainting using the preservation objective generates unrealistic samples (Sec.4.1). We randomly chose 50 validation-set images (a) from 52 ImageNet bird classes and compute their segmentation masks via a pre-trained DeepLab model [39] (b). We found that using the DeepFill-v1 inpainter to inpaint the foreground region (i.e. our "deletion" task) yields realistic samples where the object is removed (d). In contrast, using the inpainter to fill in the background region (i.e. "preservation" task) yields unrealistic images whose backgrounds contain features (e.g. bird feathers or beaks) unnaturally pasted from the object (c).

LIME-G <u>outperformed</u> LIME (top-10 cases)   LIME-G <u>underperformed</u> LIME (top-10 cases)

Real   LIME   LIME-G   Real   LIME   LIME-G

Fig. S4: Top-10 cases where the LIME-G outperformed (left) and underperformed (right) LIME on the object localization task (IoU scores). From left to right, on each row: we show a real image with its ground-truth bounding box, LIME heatmap & its derived bounding box, LIME-G heatmap & its derived bounding box. See https://drive.google.com/drive/u/2/folders/10JeP9dpuoa0M16xe2FloBEWajQ7PNKSX for more examples of the LIME and LIME-G IoU results.

Top-10 cases where SP-G <u>outperformed</u> SP   Top-10 cases where SP-G <u>underperformed</u> SP

Real   SP   SP-G   Real   SP   SP-G

Fig. S5: Top-10 cases where the SP-G outperformed (left) and underperformed (right) SP on the object localization task (IoU scores). From left to right, on each row: we show a real image with its ground-truth bounding box, SP heatmap & its derived bounding box, SP-G heatmap & its derived bounding box. In the cases where SP-G has a lower IoU score than SP (right panel), we observed the heatmap localizes some unique features of the object as compared to the images in the top cases where the heatmap covers the entire image. See `https://drive.google.com/drive/u/2/folders/1XJ6M0AMHxZrXxLLw6m3Bx7sjvsyqN6JC` for more examples of the SP and SP-G IoU results.

(a) α vs Localization error for ImageNet

(b) α vs Saliency Metric for ImageNet

(c) α vs Saliency Metric for Places365

Fig. S6: Localization error (a) and saliency metric (b, c) performance of different attribution methods on a held-out set of 1000 images for different α threshold values. For each method, we search for the optimal α value on this held-out set and use the subsequent threshold for computing the scores on the 2000 images in the object localization and saliency metric experiments in Sec. 4.3.

Fig. S7: Random intermediate perturbation samples by SP and SP-G on the same image from the nail class in ImageNet. SP-G drops the target-class probability only when the patch cover a major area of the nail (e.g. the center 0.394-probability sample in the bottom panel). This figure is a zoom-in version of the samples in Fig. 7.

Fig. S8: Qualitative evidence supporting the LIME-G vs. LIME sensitivity experiment in Sec. 4.4. For both LIME and LIME-G, per image, we compute an average SSIM score across all 10 pairs of 5 heatmaps. We then take the difference between LIME-G and LIME and sort them in the descending order. This steam locomotive image is a random image from the top-100 ImageNet-S cases where LIME-G outperformed LIME. **Top four rows:** Here, we compare pairs of LIME vs. LIME-G perturbation samples that were created from the same random superpixel masks. LIME-G samples cause large probability drops only when some discriminative feature is removed from the image and thus results in more localized heatmaps. **Bottom two rows:** 5 heatmaps by LIME and LIME-G, each from a random seed. While LIME-G heatmaps are more consistent, LIME heatmaps is noisy and varies. See Fig. S9 and Figs. S10-S11 for similar observations in ImageNet-S and Places365-S dataset respectively. See `https://drive.google.com/drive/u/2/folders/1sKWig4Xk5Pm50kdONdAS9SkiTBhJRAkw` for more examples.

Fig. S9: Here, we show the same figure as Fig. S8 (see its caption) but for another random image among the top-100 ImageNet-S cases where LIME-G outperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1sKWig4Xk5Pm50kdONdAS9SkiTBhJRAkw` for more examples.
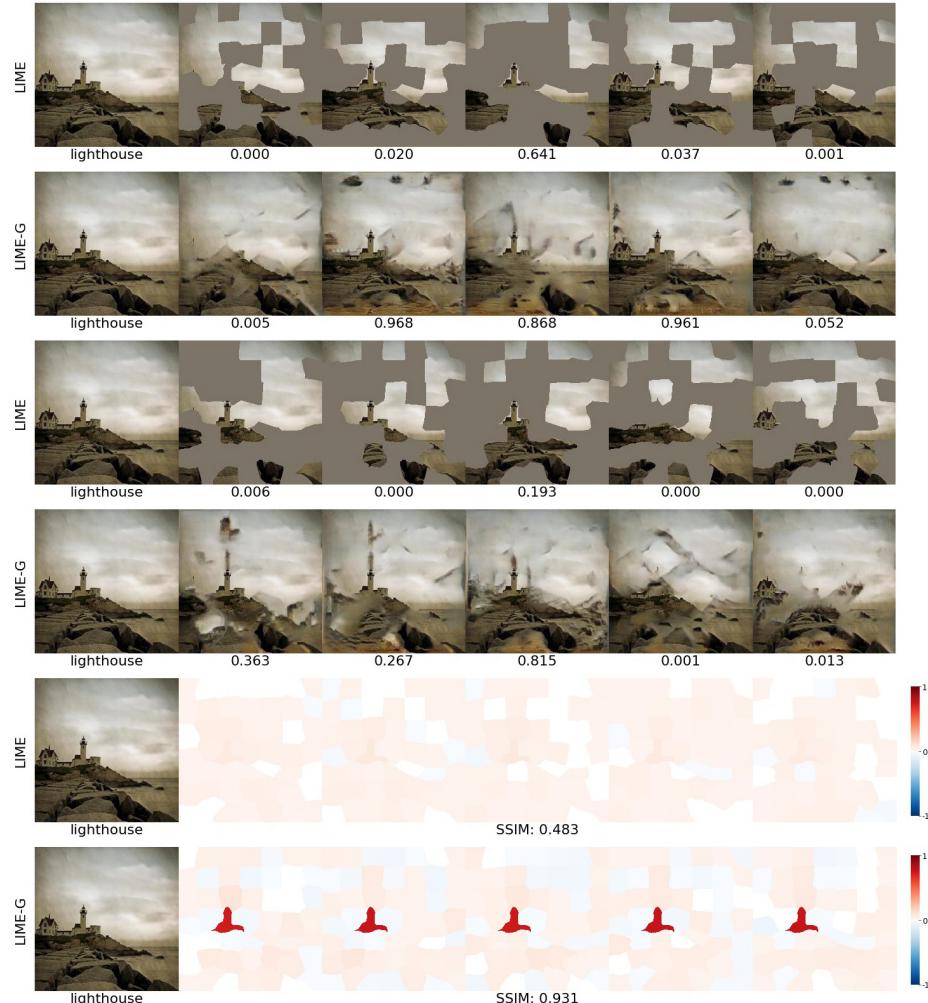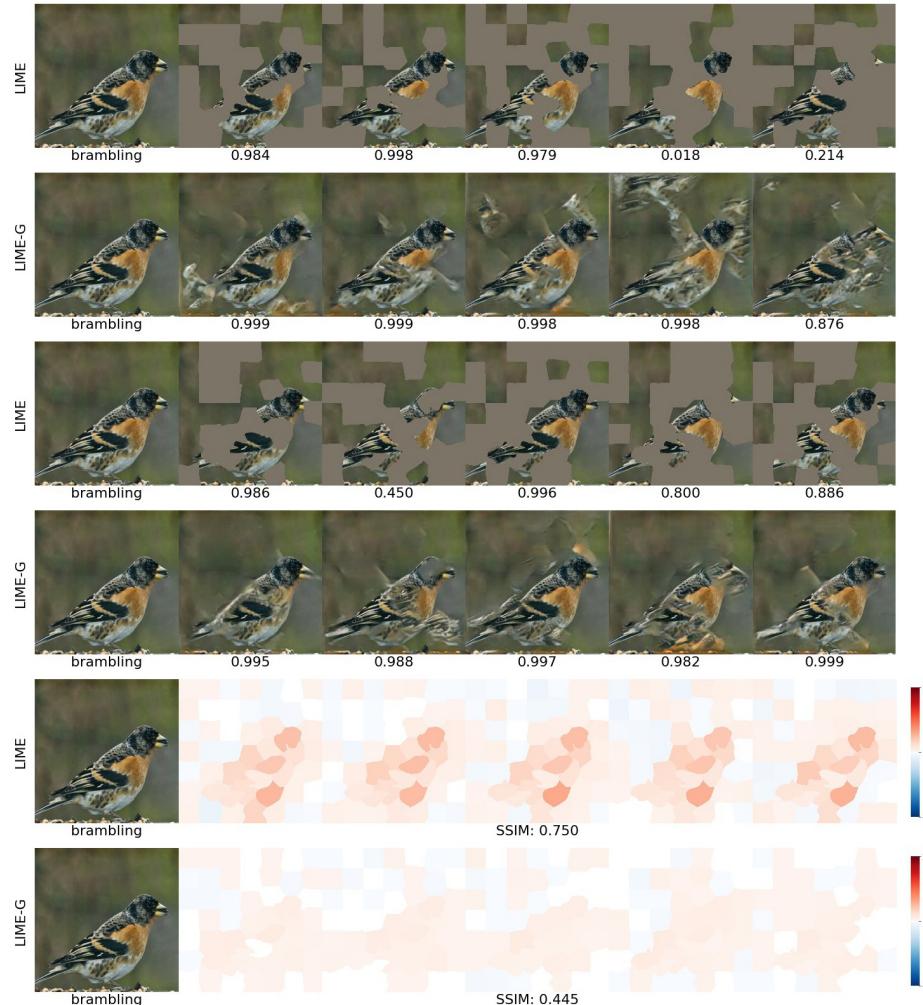
Fig. S10: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 Places365-S cases where LIME-G outperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1aXyDFBq0HlcI0kQJpJyspNf2rtwLj35Z` for more examples.

Fig. S11: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 Places365-S cases where LIME-G outperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1aXyDFBq0HlcI0kQJpJyspNf2rtwLj35Z` for more examples.
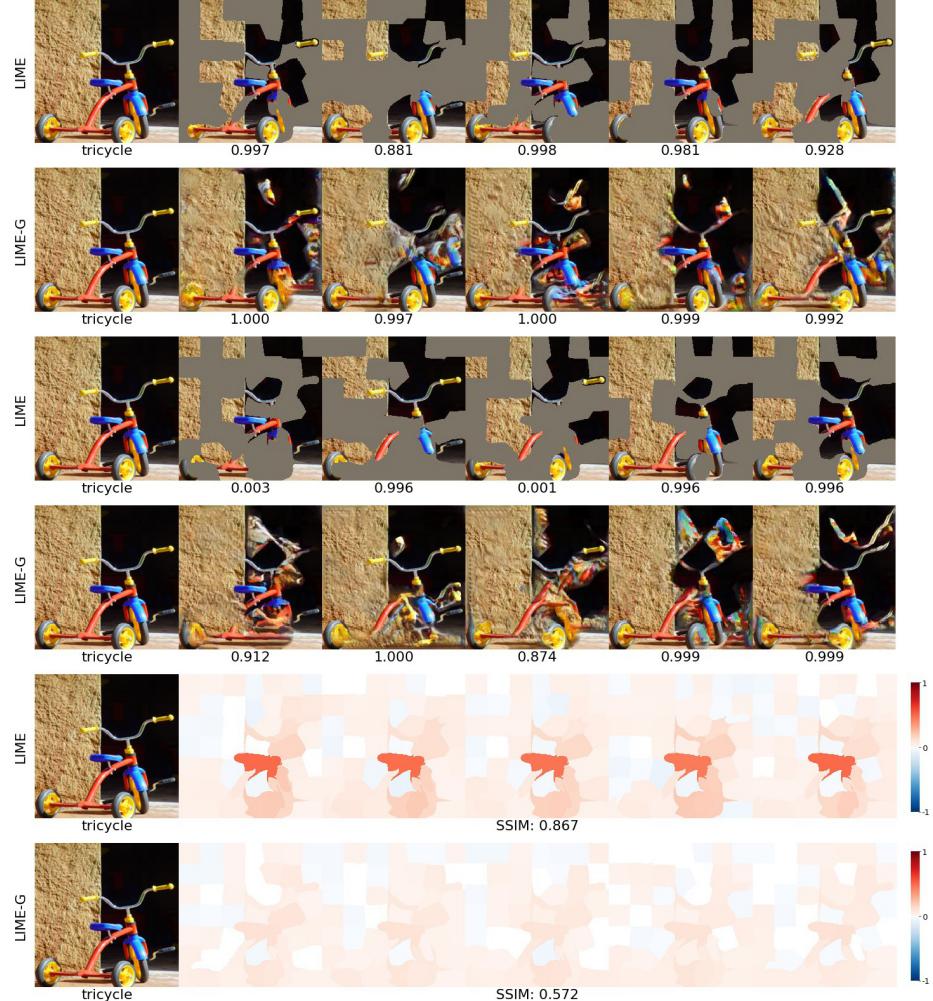
Fig. S12: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 ImageNet-S cases where LIME-G underperformed LIME on the SSIM similarity metric. LIME-G samples remain at high target-class probabilities and therefore produced heatmaps that are more sensitive than those of LIME. Similar observations can be found in Fig. S13 and Figs. S14-S15. See `https://drive.google.com/drive/u/2/folders/1sKWig4Xk5Pm50kdONdAS9SkiTBhJRAkw` for more examples.

Fig. S13: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 ImageNet-S cases where LIME-G underperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1sKWig4Xk5Pm50kdONdAS9SkiTBhJRAkw` for more examples.
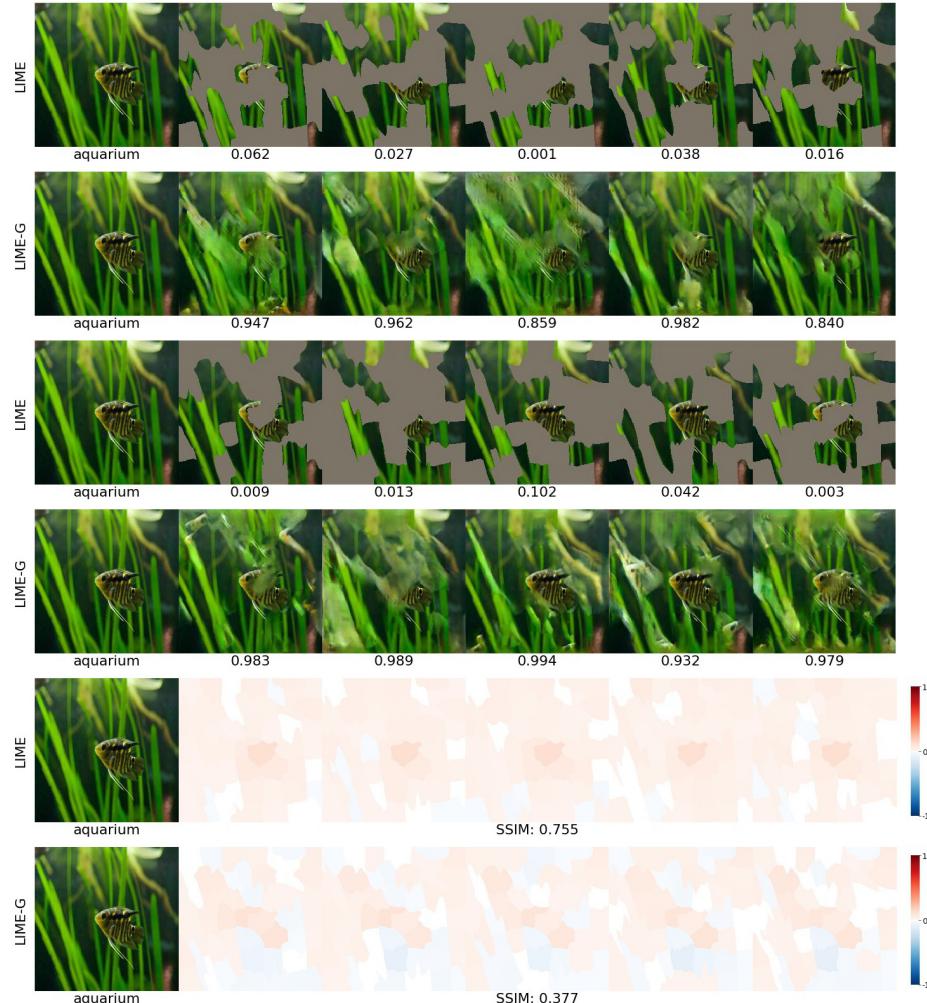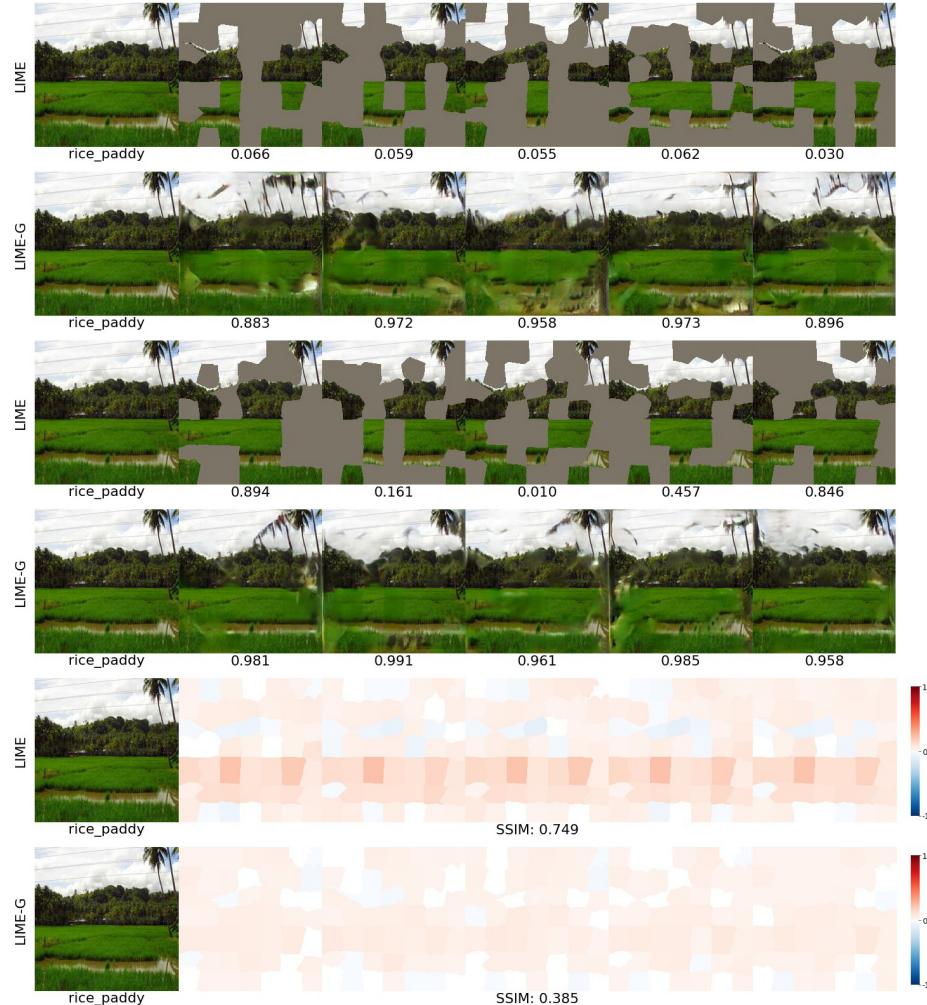
Fig. S14: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 Places365-S cases where LIME-G underperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1aXyDFBq0HlcI0kQJpJyspNf2rtwLj35Z` for more examples.

Fig. S15: Here, we show the same figure as Fig. S8 (see its caption) but for a random image among the top-100 Places365-S cases where LIME-G underperformed LIME on the SSIM similarity metric. See `https://drive.google.com/drive/u/2/folders/1aXyDFBq0HlcI0kQJpJyspNf2rtwLj35Z` for more examples.

(a) LIME histogram distribution is skewed

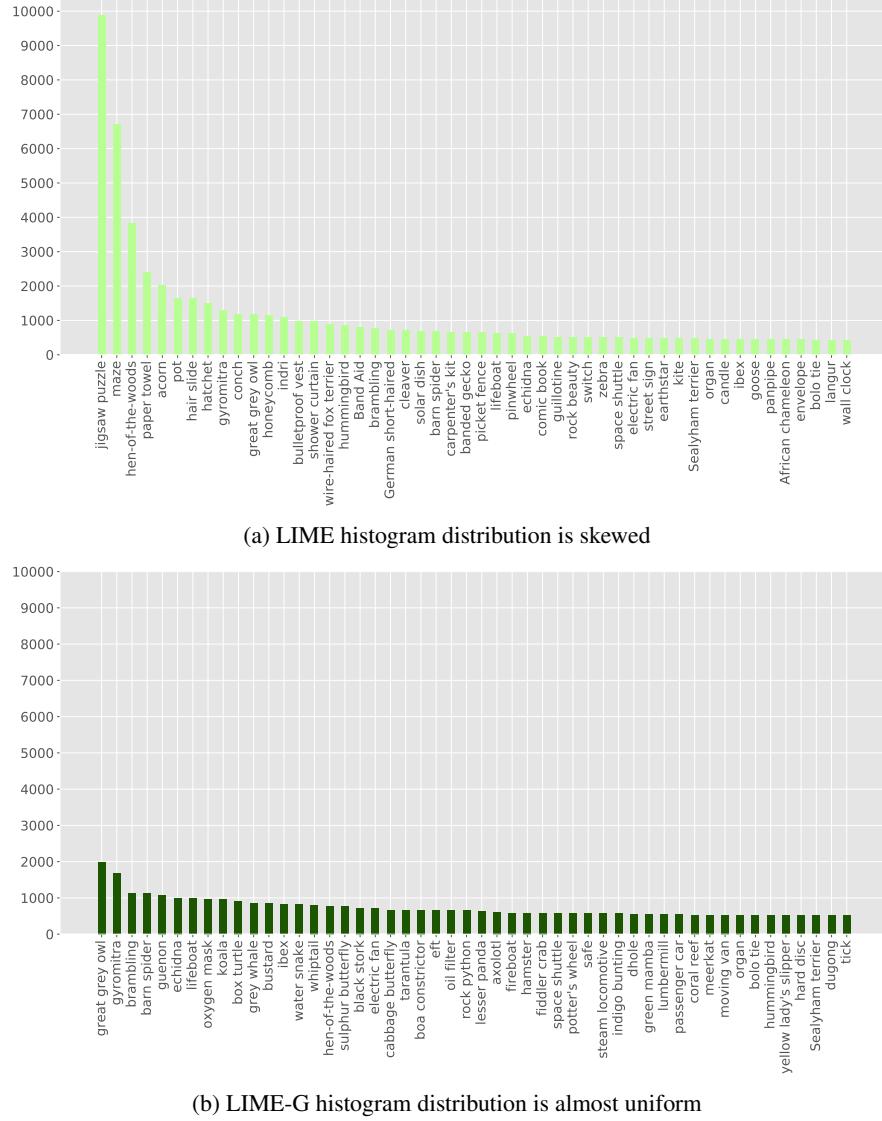(b) LIME-G histogram distribution is almost uniform

Fig. S16: We ran LIME and LIME-G on 200 images, each run has 500 intermediate perturbation samples. Here, for LIME (a) and LIME-G samples (b), we show a histogram of the top-1 predicted class labels for all 200 runs ×500 samples = 100,000 images. The set of 200 images comprises of cases where LIME-G outperformed (100 images) and underperformed (100 images) LIME on the SSIM sensitivity metric (Sec. 4.4). LIME perturbed samples are highly biased towards few jigsaw puzzle, maze classes (top panel), which is somewhat intuitive given the gray-masked images (see Figs. S8–S13). In contrast, the histogram of LIME-G samples are almost uniform. **x-axis:** For visualization purposes, we sorted the top-1 labels and showed only first 50 labels.

(a) SSIM   (b) Pearson correlation of HOG features (c) Spearman rank correlation
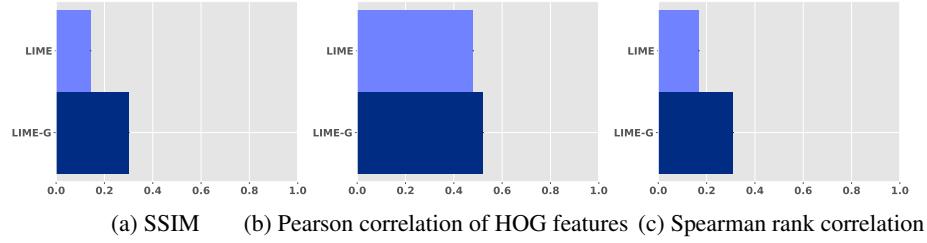
Fig. S17: Bar plots comparing the LIME vs. LIME-G robustness (higher is better) across two different numbers of superpixels $S \in \{50, 150\}$ under three different similarity metrics: SSIM (a), Pearson correlation of HOG features (b), and Spearman rank correlation (c). For each image in 1000 random ImageNet-S images, we produced a pair of heatmaps by running LIME (light-blue) or LIME-G (dark-blue) with two different numbers of superpixels $S \in \{50, 150\}$. Each bar shows the mean similarity across all 1000 heatmap pairs. LIME-G is consistently more robust than LIME, specifically by $\sim$200% under the SSIM (a) and Spearman rank correlation (c).