2024 IEEE International Conference on Image Processing (ICIP) | 979-8-3503-4939-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICIP51287.2024.10647374

SALIENCY AS A SCHEDULE: INTUITIVE IMAGE ATTRIBUTION

Aniket Singh, Anoop Namboodiri

International Institute of Information Technology, Hyderabad
Center for Visual Information Technology
Gachibowli,Hyderabad

ABSTRACT

Image Attribution seeks to reveal the importance of image regions in the classifier's final decision. Of the various ways to tackle this problem, the optimization based perspective is particularly intuitive: It applies the attribution as a mask on the image and reduces the attribution task to a loss, that can be optimized using gradient descent. Previous work has considered the goal as searching for the *single best mask*. Under this setup, however, there is a tendency towards trivial solutions of large masks with reduced discernment of the relative importances of regions. This has typically required auxiliary loss terms to control the area of the mask, however, their strength relative to the primary loss needs to be tuned. We challenge this necessity, by re-imagining attribution as an ordering of pixels according to importance. This ordering may be interpreted as a *schedule* which determines which locations get seen earlier and which later, allowing us to create a trajectory of masks from completely OFF to completely ON. We optimize through this sequence of masks of "all" areas and not just a single mask as in previous methods. We explore this setting, which we dub *Saliency-as-Schedule* (SaS), and demonstrate its effectiveness through experiments in a variety of settings, involving multiple datasets and CNN architectures. Further, we also propose a novel attribution task, *feature saliency*, where we use SaS to rate the influence of image regions on the *intermediate* feature maps of a CNN, and not just the class logit. Our findings suggest that SaS is a promising direction for the attribution problem. Our code will be available at https://github.com/tumbleweed/SaliencyasSchedule

Index Terms— Image Attribution, GradCAM, Gradient Descent, Optimization, Saliency

1. INTRODUCTION

Modern deep learning techniques have been remarkably successful, leading to a paradigm shift away from hand designed features towards a largely data-driven methodology. With this "hands-off" approach of letting the model learn what it will from the data, there is an open question about what eventually is learned. This question is addressed by various approaches such as feature visualization [1] [2] [3] [4], concept vectors [5], and image attribution [6] [7] [8] [9] [10].

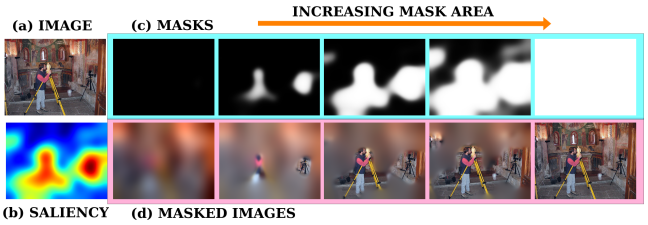

Fig. 1. Overview of Saliency as Schedule: (a) An image containing 2 Tripods. The saliency (b) is interpreted as a schedule or priority order over locations to (c) create a cascade of masks of increasing area. The trajectory is non-linear, the masks show high contrast, and larger masks contain smaller ones. (d) Masked images corresponding to the masks

Of interest here is *attribution*, which seeks to highlight prominent areas in the image that influence the classifier's decision. There are many ways to pose attribution, from restructuring backpropagation [9] [11], to repeatedly querying the network with altered inputs [12] [13]. An interesting subgroup is *Optimization based Attribution* (OAttr), which poses it as an optimization problem to be solved using gradient descent. OAttr searches for a mask for the class of interest, where the effectiveness of the mask is measured by its ability to preserve regions of the image that lead to high CNN scores. This formulation is attractive due to its intuitive way of quantifying what is a "good" mask, in contrast to heuristic or rule based techniques. Further, the success of gradient descent for deep learning in general, as well as for the adjacent explanation technique of feature visualization, makes this a setup worth exploring.

However, a challenge associated with optimization algorithms in general, is that of trivial solutions. In OAttr, this manifests as large masks that may retain surplus area in an attempt to greedily influence the CNN score. This is typically addressed with an auxiliary term to keep the mask area small: Meaningful Perturbations (MP) [7], IGOS [14], IGOS++ [15], use $\mathcal{L}_1$ or $\mathcal{L}_2$ penalties on the mask. The strength of this term, relative to the primary loss, is a hyperparameter to be fixed. Extremal Perturbation (EP) [16] simplifies the mask penalty, making the desired object size explicit as a hyperparameter, and regressing the mask's area towards it.

Here too, however, the size of the object being searched for must be tuned to the dataset.

In contrast to these works which optimize for a single mask, we re-imagine saliency as a trajectory of masks from completely ON to completely OFF (figure 1). Such a path determines a *schedule* or a priority order, in which pixels from the original image are revealed to CNN. The goal of our scheme, dubbed *Saliency as Schedule* (SaS), is to find the best such trajectory. In this process, we optimize through masks of "all" areas. Further, the design of the path is compositional, where larger masks contain smaller masks (figure1), conveying the inductive bias that a region considered important should continue to be important when seen together with other parts of the image.

We demonstrate the effectiveness of this approach, by quantifying performance on a variety of experimental settings, involving multiple model architectures and datasets. We compare against several well-known attribution schemes, both quantitatively, using the *deletion game* benchmark, as well as qualitatively. Finally, we present a new attribution task using SaS, *feature saliency*, to determine how image regions influence intermediate feature maps of the classifier. Our findings suggest conceptualizing saliency as a schedule is an promising direction for attribution.

2. RELATED WORK

The techniques most related to our work are the Optimization based Attribution (OAttr) schemes. This line of work arguably begins with Meaningful Perturbations (MP) [7], which formalizes the original optimization problem for saliency, though the usage of optimization for explanation can be traced back to [17], [18] and [1]. Subsequently, this domain has seen the development of methods like IGOS [14], IGOS++ [15] and Extremal Perturbation (EP) [16]. OAttr methods seek to discover the attribution as a mask for the image using gradient descent. The optimization is driven by an objectness loss L_o, that measures the effect of the masked image on the CNN scores. However, MP identified challenges of the adversarial phenomenon [19] & trivial solutions of large masks, that cannot be circumvented by the objectness term alone. Auxiliary terms L^i_{aux} are needed to meet all or some of these issues, leading to the typical attribution loss L_{attr} as:

$$L_{attr} = L_o + \sum_i \lambda_i L^i_{aux} \tag{1}$$

where λ_i are the weights for auxiliary terms.

While the adversarial phenomenon is addressed variously via a smoothness auxiliary term L_{adv} (MP, IGOS & IGOS++) or via a smoothness operator (EP), all popular OAttr methods address large masks via some type of mask compactness term, L_{size}. An issue with such auxiliary terms is that they must be balanced against the main loss and each other, and their weights must be tuned to the dataset at hand.

Apart from operating within this common framework, these methods differ from each other in various aspects. IGOS [14] incorporates ideas from Integrated Gradients [8], leveraging gradients of a series of images between a baseline and the masked image.IGOS++ [15] simultaneously optimizes the insertion and deletion modes (see section 3, Loss).

An alternate school of thought on attribution is the Class Activation Maps (CAMs). Beginning with the success of GradCAM [6], works such as ScoreCAM [12], GroupCAM [20], etc. propose different rules for computing the saliency. One of the early works on attribution for deep networks [18], demonstrated the ability of input gradients to locate the object of interest in the image. Subsequent works such as Guided-Backpropagation [11] etc. develop this theme further. Integrated Gradients (IG) [8] traces its foundations to game theoretic ideas. IG creates a linear path of images from a baseline to the original image, and accumulates the class gradients on the entire path to produce the final saliency.

3. SALIENCY AS SCHEDULE

Our approach, Saliency-as-Schedule (SaS), optimizes the importance map for class c for a given image I. The machine learning model being investigated is the CNN $f(\cdot)$, whose score (pre-softmax logit) for class c is given by $f_c(\cdot)$. The CNN input size and the size of I are (H, W).

The entity being optimized in our framework is the *pre-mask* $\overline{m}$, of size (H, W), and whose pixels take on values in the range $[-\infty, \infty]$. The aspect that distinguishes our approach from previous OAttr schemes is that we utilize $\overline{m}$ to create a trajectory of masks $\{m_k\}_{1..K}$ that are used in the optimization.

Smoothness Previous works [7] [16] have noted that parameterizing the saliency map as a collection of independent pixels is susceptible to the adversarial phenomenon [19]. For attribution, this phenomenon leads to high-frequency artefacts in the mask, where the importance values can vary greatly within small neighborhoods. Such masks do not adhere to the common-sense notion of saliency, which requires that a reasoning system should attend to *regions*, which are contiguous areas of the image, rather than individual pixels. To avoid the adversarial phenomenon, we wish to have an inductive bias towards smooth masks. We impose this by smoothing $\overline{m}$ with a Gaussian kernel to yield $\texttt{Smooth}(\overline{m})$. However, as observed in [16], linear low-pass filtering leads to results with gentle transitions between high and low values. To enhance contrast, we can perform a subsequent $\texttt{SoftMaxPool}$ of $\texttt{Smooth}(\overline{m})$, to give sharper transitions between values, while retaining non-sparse gradients (unlike $\texttt{MaxPool}$):

$$\widetilde{m} = \texttt{SoftmaxPool} \circ \texttt{Smooth}(\overline{m}) \tag{2}$$

(where $\circ$ represents function composition).

Creating a mask trajectory: We interpret $\widetilde{m}$ as defining a *schedule* or priority order over locations. Our intuition is that of a spotlight on the image that slowly grows to reveal its different parts. An effective spotlight would reveal the most class-relevant areas before the irrelevant regions. We concretize this intuition by creating a trajectory of K binary

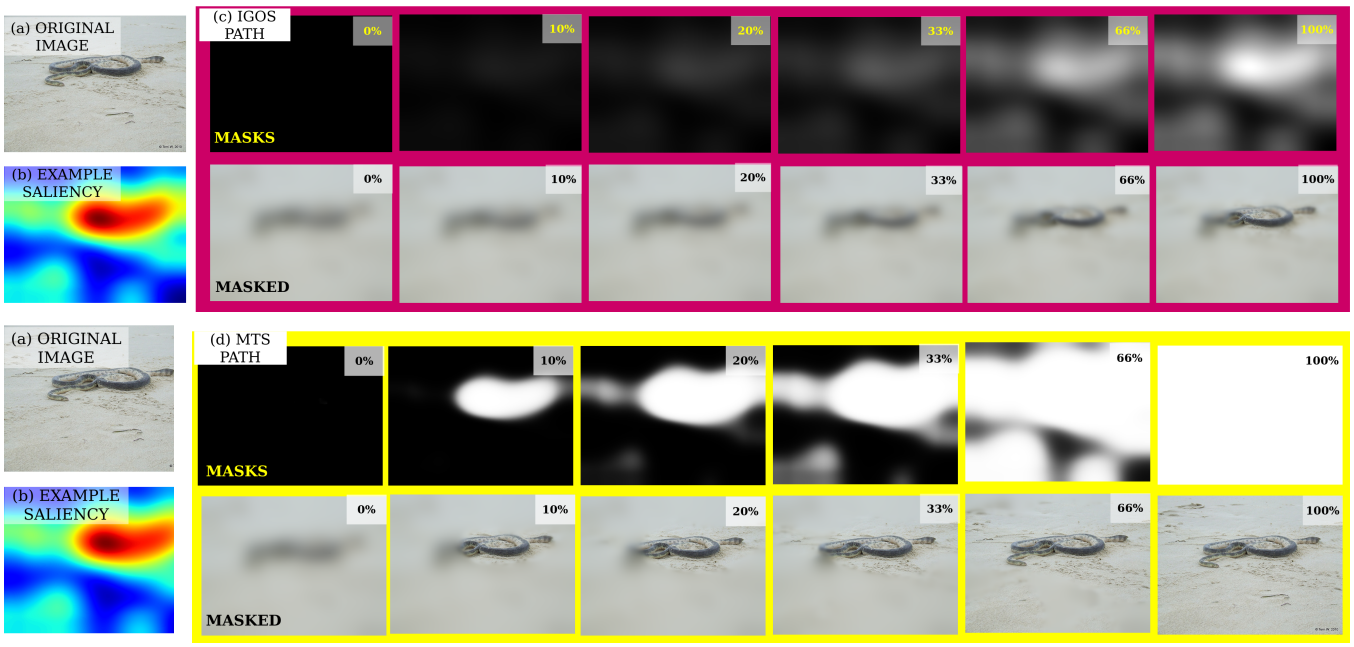

Fig. 2. Comparison of masking modules of IGOS [14]/IGOS++ [15] and SaS: In both techniques, the attribution (b) is used to produce a cascade of masks. In IGOS/IGOS++ (c), this cascade is a linear interpolation between the OFF mask and the attribution. In SaS (d) a non-linear procedure is used to create the cascade between the OFF mask and the ON mask. SaS masks are sharper, and upon application either strongly suppress or retain pixel information. In contrast, intermediate masked images from IGOS/IGOS++ are blurred

masks $\{m_k\}_{1..K}$ from the schedule $\widetilde{m}$, where any member m_k has the most important $A_k = \frac{100k}{K}\%$ of the pixels as ON.

However, as our optimization based approach requires gradients and truly binary masks are non differentiable, we relax this requirement, to work with soft masks m_k whose pixels take on values in the range $[0, 1]$. In order to generate the trajectory $\{m_k\}$ from $\widetilde{m}$, we use the sigmoid operator σ:

$$m_k = \sigma(\widetilde{m} - t_k) \tag{3}$$

where t_k is a scalar in $[-\infty, \infty]$. Notice that as $t_k \to -\infty$, all the values in m_k turn completely ON, i.e. $m_k \to 1$, and as $t_k \to \infty$, m_k has all values as completely OFF, i.e. $m_k \to 0$. At intermediate $t_k \in [-\infty, \infty]$, the elements of m_k take on values between 0 & 1.

Since we wish to create a trajectory $\{m_k\}$, who masks are equally separated in area, we introduce the `Size` operator to measure the mask area using the $\mathcal{L}_2$ norm:

$$\texttt{Size}(m) = \|m\|_2^2 \tag{4}$$

We use the `Size` operator to place a constraint on m_k:

$$\frac{\texttt{Size}(m_k)}{HW} = \frac{k}{K} \tag{5}$$

We use this constraint to define an optimization problem in t_k:

$$t_k = \underset{\hat{t}_k}{\operatorname{argmin}} \left\| \frac{100\texttt{Size}(\sigma(\widetilde{m} - \hat{t}_k))}{HW} - A_k \right\| \tag{6}$$

We solve for the $\{t_k\}$ efficiently by using a batched GPU implementation of the Levenberg-Marquardt algorithm. Note, as the schedule evolves during optimization, the $\{t_k\}$ that induce equi-spaced areas of $\{m_k\}$ must also be recomputed. In practice, we observed faster convergence when $\hat{t}_k$ was initialized by the A_k-th *reverse percentile* of $\widetilde{m}$.

Figure 2 compares the masks generated by this approach, with the masking mechanism of IGOS & IGOS++. IGOS creates masks between the completely OFF mask and the saliency, while SaS creates masks of all areas between the completely OFF and the completely ON masks. Thus, under our formalism, IGOS optimizes over an incomplete path. Secondly, IGOS performs a linear interpolation between the end points, while SaS creates a non-linear path. This leads to IGOS masks taking on a diffuse appearance, leading to masked images that are blurred globally. SaS masks, on the other hand, are sharper, which leads to masked images where regions are either almost completely suppressed or almost completely retained.

Applying the mask:To implement the masking operator, we use the blur pyramid described in [16], where the mask is used to interpolate between blur levels of the image I. Lower values in the mask correspond to higher blurr and vice versa. We omit the technical details and represent the image created by applying the mask m to image I as $\texttt{Mask}(\texttt{I}, \texttt{m})$. Interested readers may refer to [16]

Loss If we wish to understand which areas of an image are relevant for a learning agent, we can selectively reveal small

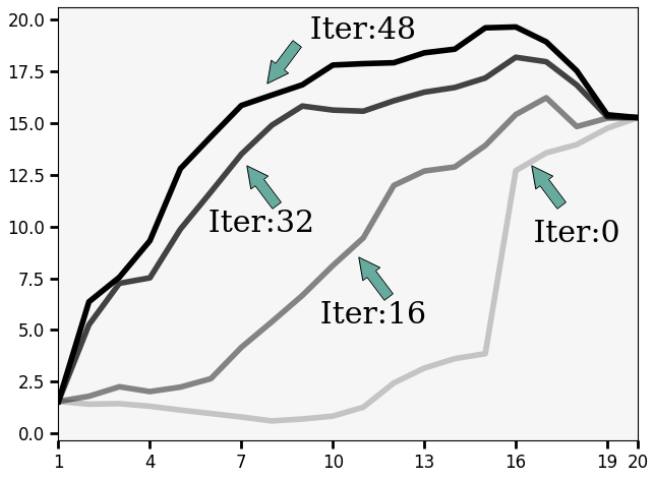

Fig. 3. Score profiles of masked images: Each line represents the CNN scores predicted for masked images at a particular iteration. All graphs share common starting and ending points, representing the OFF & ON masks respectively. As optimization proceeds, the scores increase on average, and also take on a "concave" profile. See section 3

parts of the image and observe where the agent's recognition improves. Alternately, we can also test for importance by selectively hiding parts of the image and observing where the agent's accuracy falls. Thus far we have described our framework with the former strategy, known as *Insertion*, although the latter, known as *Deletion* is equally applicable.

When using the Insertion mode within SaS we progressively reveal regions of the image starting from the most important (as deemed by the schedule), and culminating in the entire image. In this setting, the objective L_I is to maximize the CNN class score on the masked images. Ideally if the important regions are correctly identified, they are part of more masks in $\{m_k\}$, causing the CNN score to go up. In Deletion, the objective L_D, is to minimize the CNN output for the masked images. If the important regions are identified correctly, they are absent from more masks, leading to large dips in the CNN score. Note, Deletion can be implemented simply by masking by $\{1 - m_k\}$ instead of m_k. We can create a third mode, *Both*, by combining the Deletion and Insertion modes. In this setting, we create masks of both the Deletion and Insertion type, and compute the loss L_{I+D} as the sum of the Insertion and Deletion losses.

$$\begin{aligned} \mathcal{L}_I &= -\sum_k f_c(\texttt{Mask}(I, m_k)) \\ \mathcal{L}_D &= \sum_k f_c(\texttt{Mask}(I, 1 - m_k)) \\ \mathcal{L}_{I+D} &= \mathcal{L}_I + \mathcal{L}_D \end{aligned} \tag{7}$$

When running SaS, we use one of the losses in equation 7 as the objective for performing gradient descent on the pre-mask $\widetilde{m}$. Note the absence of any auxiliary loss on the mask area: The simplicity of the setup requires only the CNN loss. The final attribution m^* for visualization is a normalized version of $\widetilde{m}$:

$$m^* = \frac{\widetilde{m} - \texttt{min}(\widetilde{m})}{\texttt{max}(\widetilde{m}) - \texttt{min}(\widetilde{m})} \tag{8}$$

In figure 3, we consider the working of SaS for a single sample and observe the class scores of the masked images across the various stages of optimization. The loss mode is Insertion. It can be observed, as optimization proceeds the score profiles tend to rise, tending towards a concave or hump-like shape. This tendency towards concavity can be explained as follows: As optimization proceeds the earlier/smaller masks of the trajectory $\{m_k\}$ succeed in introducing class-relevant regions, leading to a consistent rise in the score. Once the class-relevant regions have been exhausted by the earlier masks, the larger/later masks must introduce the remaining irrelevant regions, leading to a drop in scores. The profile terminates in the score of the original image (corresponding to the fully ON mask). Note, for Deletion this tendency would have been towards convexity.

4. EXPERIMENTS

Experimental Setup We run our quantitative experiments on 3 datasets: MNIST , CIFAR-10 & CIFAR-100. While MNIST contains the 10 digit classes with grayscale images of size of 28×28, CIFAR-10 and CIFAR-100 consist of color images of size 32×32, with 10 and 100 classes respectively. We run our experiments on 2 different types of architectures, VGG16 and ResNet8. These networks take input of size 32×32, for which we resize the MNIST images. To train our models, we rely on open-source implementations provided in [21] and [22]. The test accuracies for these networks are given in table 1. We also provide qualitative results on samples from the Imagenet dataset, which consists of color images from 1000 classes. For these results we run SaS on a VGG16 network, whose input size is 224×224

Table 1. Test performance of the networks utilized for the quantitative study. For details, see section 4

	CIFAR-10	**CIFAR-100**	**MNIST**
ResNet8	93.12 %	75.17 %	99.22 %
VGG16	93.75 %	72.53 %	99.75 %

Implementation Details For all datasets in our experiments, we generated 20 masks from the schedule. For MNIST, CIFAR-10, CIFAR-100, we performed optimization for 200 iterations using a simple gradient descent based optimizer, using the combination of Deletion and Insertion loss modes (Both mode) (see section 3). For Imagenet attribution, we optimize for 800 iterations. Unlike previous OAttr schemes which optimize for a smaller saliency object than the image size, our iterate, the pre-mask $\overline{m}$ (see section 3), is of the same size as the image.

Deletion Game In order to quantify the performance of SaS against other techniques, we use the Deletion Game

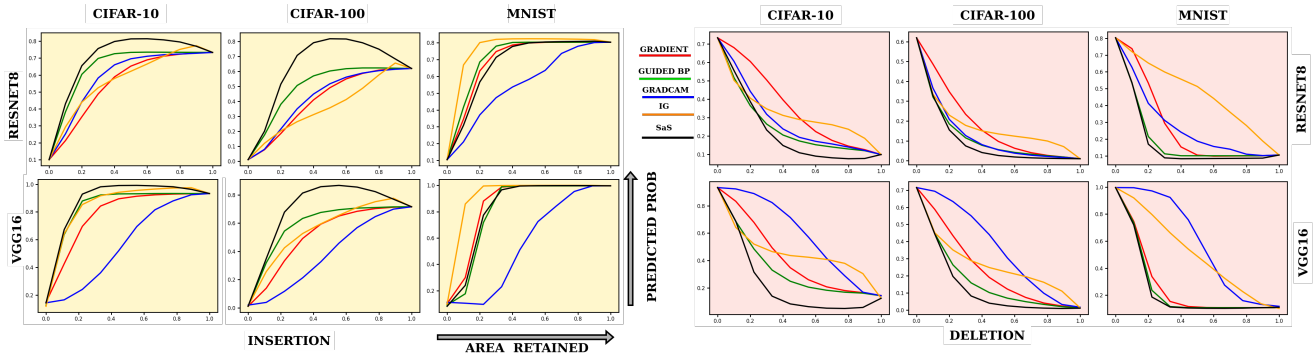

Fig. 4. Deletion and Insertion Game Graphs for various experimental settings: For Insertion, higher graphs (towards top-left) imply superior performance, while for Deletion lower (towards bottom-right) is better. SaS (black line) outperforms other methods in most settings.See section 4

Table 2. Deletion/Insertion Game scores across 3 different datasets (MNIST, CIFAR-10, CIFAR-100) and 2 different architectures (ResNet8 and VGG16) for Saliency-as-Schedule (SaS) and competing methods. Best result in bold, second best underlined. SaS outperforms competitors in most settings. See section 4

Dataset	MNIST						CIFAR10						CIFAR100					
Arch	ResNet8			VGG16			ResNet8			VGG16			ResNet8			VGG16		
Method	Ins. ↑	Del. ↓	Overall ↑	Ins. ↑	Del. ↓	Overall ↑	Ins. ↑	Del. ↓	Overall ↑	Ins. ↑	Del. ↓	Overall ↑	Ins. ↑	Del. ↓	Overall ↑	Ins. ↑	Del. ↓	Overall ↑
Guided BP	0.691	0.215	**0.476**	0.798	0.268	0.5210	0.629	0.264	0.365	0.816	0.347	0.469	0.486	0.136	0.3410	0.574	0.183	0.391
Gradient	0.675	0.284	0.391	0.827	0.285	0.542	0.544	0.363	0.181	0.761	0.418	0.343	0.405	0.191	0.214	0.504	0.247	0.257
IG	**0.738**	0.474	0.264	**0.899**	0.512	0.386	0.563	0.332	0.231	0.8210	0.459	0.370	0.365	0.188	0.177	0.543	0.273	0.2610
GradCAM	0.548	0.290	0.258	0.555	0.580	-0.025	0.577	0.293	0.284	0.562	0.581	-0.019	0.419	0.143	0.276	0.377	0.365	0.012
SaS (Ours)	0.663	**0.201**	0.463	0.808	**0.259**	**0.549**	**0.6710**	**0.236**	**0.443**	**0.858**	**0.243**	**0.615**	**0.609**	**0.119**	**0.4810**	**0.677**	**0.151**	**0.526**

benchmark [13]. The benchmark bears similarity to the insertion and deletion modes discussed in section 3, as it creates K masks from the attribution, with the k^{th} mask retaining $A_k = \frac{100k}{K}\%$ of the most important locations. These masks are applied to the image, and the CNN scores are measured. A notable difference from the soft masks required for OAttr methods like SaS, however, is the usage of *truly binary* masks.

Similar to the modes used in the optimization in section 3, the benchmarking can also be run as a Insertion or Deletion test. In the former better attributions elicit higher scores from the CNN, and for the latter, lower is better.

With regards to the masking operator used for benchmarking, it was recently observed [23] that methods like representing the removed values with a fixed constant (like 0 or the median of the image) [13], or a highly blurred version of the image [14] may lead to disagreeing rankings between evaluation game modes. To mitigate this we use the Noisy Linear Imputation scheme of [23].

Figure 4 plots the deletion/insertion performance across ratios of mask pixels removed/retained. In our tests we use 11 binary masks, uniformly increasing in area from 0% to 100%. For insertion, graphs towards the top-left are better, and for deletion, towards the bottom-right. The generally better performance of SaS in most scenarios can be observed.

For either game, the scores across the areas can be averaged to yield a single number summary. To further consolidate deletion and insertion performance into a single number, we use the *overall* score which is the difference between the insertion and the deletion score. Table 2 shows that SaS outperforms several well-known methods such as Integrated-Gradients [8], GradCAM [6], Guided Backpropagation [11] and Gradient [18] on most experimental settings: Out of the 18 measurement scenarios, it falls behind in just 3, involving MNIST. This behaviour is also reflected in the insertion/deletion graphs in figure 4 , and is interesting for further study.

Qualitative Results Figure 5 shows some qualitative comparisons on a VGG16 network trained on Imagenet. It can be seen that SaS-Insertion performs favorably against several contemporary techniques such as IGOS++ [15], Extremal Perturbation [16], GradCAM [6], ScoreCAM [12] and GroupCAM [20].

Feature Saliency While Attribution methods have largely focused on class saliency, explanations of intermediate feature maps have typically been conducted by inverting them into a pre-image [1] [3]. However, as a consequence of the detailed attribution performed by SaS, we can produce saliencies for these internal features as well.

Consider that the feature computed for the image I by the classifier f at the layer of interest l is $f^l(I)$. We describe the insertion mode for feature saliency, which seeks to reveal regions of the image that reduce the disparity between the features of the masked image and the original image. Thus the best trajectory of masks rapidly cuts down the error between the masked image features and original image features. We use the standard $\mathcal{L}_2$ norm to measure disparity, which leads to

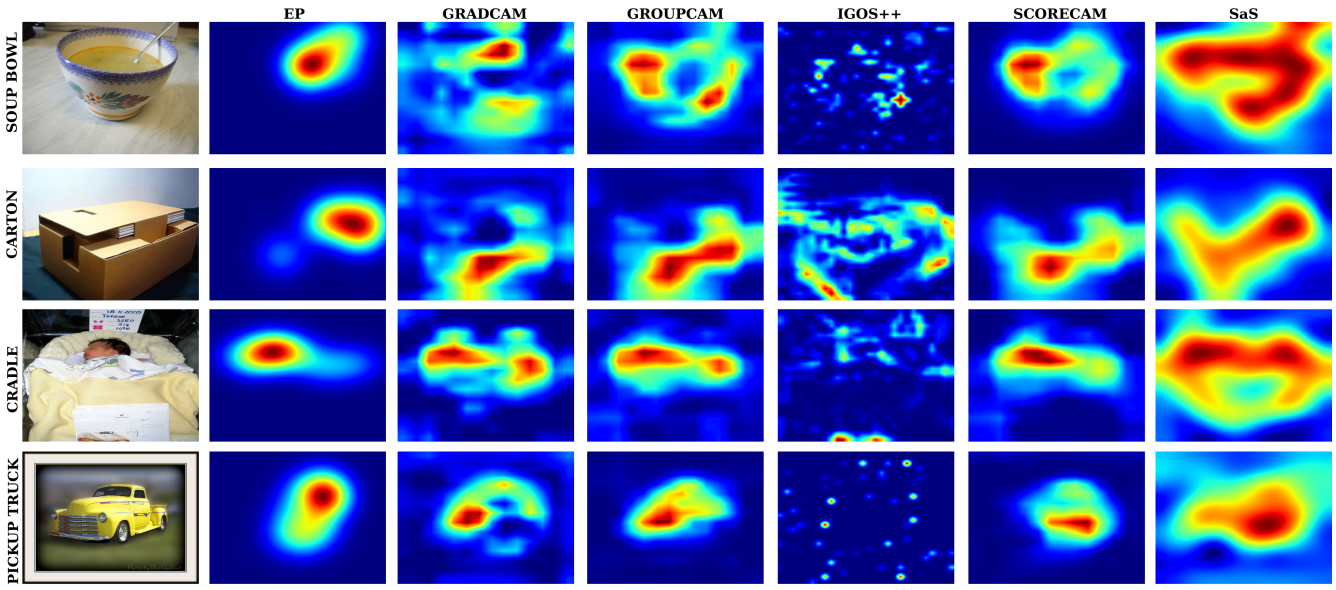

Fig. 5. Qualitative Comparison of SaS with several other methods on a VGG16 network trained on Imagenet.While Extremal Perturbation's [16] fixed area regularization results in attributions that fixate to a single region, IGOS++'s [15] area minimization term leads to very sparse saliencies. In contrast SaS is more detailed.

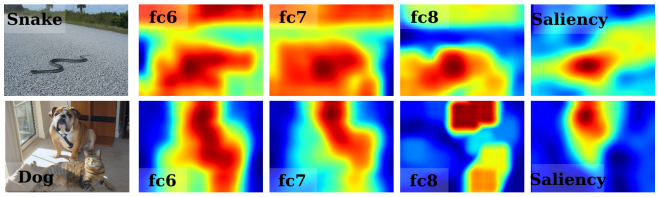

Fig. 6. Feature Saliency: At lower layers, the CNN features are influenced by regions aside from just the object: for e.g. the trees in the snake image and the cat in the dog image. As we move towards the final layer, the features become more sensitive to just the major class.

the feature saliency loss:

$$L_l = \sum_k \|f^l(I) - f^l(\texttt{Mask}(I, m_k))\|_2^2 \tag{9}$$

Note while the EP [16] paper also presents a feature attribution task, which we shall dub EP-feature-saliency, it's formulation is different from ours in three major ways: (i) For EP-feature-saliency the purpose of the saliency is to still highlight factors influencing the *final classifier decision*, while in our case we are interested in what effects an *internal map*. (ii) EP-feature-saliency produces a mask to be *applied on the internal features*, while our formulation is closer to the standard notion of saliency, as a mask to be *applied to the image* (iii) EP-feature-saliency's masks are spatially broadcasted, and seek to *select important channels*, while our masks are spatial in nature and seek to *highlight important regions*.

Figure 6 shows the feature saliencies for the 3 fully connected layers of a VGG16 network trained on Imagenet. For both scenarios considered, a trend is perceivable: the lower layer features are influenced by objects in the image aside from the major class. This influence gradually shifts away from secondary objects as we ascend the layers, finally fixating on the major class object when we pick the class logit (i.e. for class saliency). In the image with the snake, at earlier layers the features are influenced by the tree regions as well, while in the image with two objects, a dog and a cat, the early layers attend to both classes, before reducing influence of the cat class.

5. CONCLUSION

We took a fresh look at the problem of Image Attribution. Previously, a popular paradigm has been to use gradient descent to solve for the attribution as a mask that suppresses unimportant parts of the image. However, this scheme requires auxiliary constraints on the size of the mask, whose emphasis against the main loss is difficult to balance. In Contrast to this view, we re-imagined attribution not as a single mask, but as a schedule that prescribes a trajectory of masks, going from completely OFF to completely ON. Such a formulation captures the lack of knowledge about the "correct" mask area, as we optimize through masks of "all" areas, without the need for auxiliary terms for the size. We formalized this framework, which we dub Saliency-as-Schedule (SaS), and experimentally demonstrated its effectiveness on a variety of datasets and CNN architectures. Finally, we utilized it to develop a new application for attribution, *feature saliency*, which highlights the importance of image areas for the intermediate feature maps produced in the network.

6. REFERENCES

[1] A Mahendran and A Vedaldi, "Understanding deep image representations by inverting them," in *CVPR*, 2015.

[2] H Yin, P Molchanov, J M Alvarez, Z Li, A Mallya, D Hoiem, N K Jha, and J Kautz, "Dreaming to distill: Data-free knowledge transfer via deepinversion," in *CVPR*, 2020.

[3] A Singh and A Namboodiri, "Laplacian pyramids for deep feature inversion," in *ACPR*. IEEE, 2015.

[4] D Ulyanov, A Vedaldi, and V Lempitsky, "Deep image prior," in *CVPR*, 2018.

[5] B Kim, M Wattenberg, J Gilmer, C Cai, J Wexler, F Viegas, et al., "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *ICML*. PMLR, 2018.

[6] RR Selvaraju, M Cogswell, A Das, R Vedantam, D Parikh, and D Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *ICCV*, 2017.

[7] R C Fong and A Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *ICCV*, 2017.

[8] M Sundararajan, A Taly, and Q Yan, "Axiomatic attribution for deep networks," in *ICML*. PMLR, 2017.

[9] G Montavon, A Binder, S Lapuschkin, W Samek, and KR Müller, "Layer-wise relevance propagation: an overview," *Explainable AI: interpreting, explaining and visualizing deep learning*, 2019.

[10] A Singh and A Namboodiri, "Image attribution by generating images," in *ICASSP*, 2024.

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

[12] H Wang, Z Wang, M Du, F Yang, Z Zhang, S Ding, P Mardziel, and X Hu, "Score-cam: Score-weighted visual explanations for convolutional neural networks," in *CVPR workshops*, 2020.

[13] V Petsiuk, A Das, and K Saenko, "Rise: Randomized input sampling for explanation of black-box models," *BMVC*, 2018.

[14] Z Qi, S Khorram, and F Li, "Visualizing deep networks by optimizing with integrated gradients.," in *CVPR Workshops*, 2019, vol. 2.

[15] S Khorram, T Lawson, and L Fuxin, "igos++ integrated gradient optimized saliency by bilateral perturbations," in *CHIL*, 2021.

[16] R Fong, M Patrick, and A Vedaldi, "Understanding deep networks via extremal perturbations and smooth masks," in *ICCV*, 2019.

[17] D Erhan, Y Bengio, A Courville, and P Vincent, "Visualizing higher-layer features of a deep network," *University of Montreal*, vol. 1341, no. 3, 2009.

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

[19] C Szegedy, W Zaremba, I Sutskever, J Bruna, D Erhan, I Goodfellow, and R Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[20] Q Zhang, L Rao, and Y Yang, "Group-cam: Group score-weighted visual explanations for deep convolutional networks," *arXiv preprint arXiv:2103.13859*, 2021.

[21] "cifar10-fast-simple," 2021, `https://github.com/99991/cifar10-fast-simple`.

[22] "pytorch-vgg-cifar10," 2017, `https://github.com/chengyangfu/pytorch-vgg-cifar10`.

[23] Y Rong, T Leemann, V Borisov, G Kasneci, and E Kasneci, "A consistent and efficient evaluation strategy for attribution methods," in *ICML*. 2022, PMLR.