Lecture notes on explainable AI: Saliency maps

Aasa Feragen

June 5, 2023

1 Introduction

These lecture notes are associated with the lecture on saliency maps held in the course *Deep Learning in Computer Vision* held at DTU in June 2023. If you find any errors or typos, please pass them on to `afhar@dtu.dk`.

1.1 Motivation – what are saliency maps?

Saliency maps are usually defined as heatmaps that seek to give a visual answer to the fundamental question

Fundamental Question 1 *How much does each region of the image contribute to a given decision?*

Over the next few pages, we will visit a few different classical definitions of saliency maps, and discuss the extent to which they live up to this goal.

2 Definitions of saliency maps

2.1 Gradient-based saliency maps

As taught in introductory calculus, the gradient of a predictive function tells you how you should change the function's inputs in order to most efficiently increase its output. In mathematical terms, for a function $f\colon \mathbb{R}^n \to \mathbb{R}$, the gradient

$$\nabla_x f = (\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n})$$

at $x \in \mathbb{R}^n$ is a vector in $\mathbb{R}^n$ pointing in the direction of maximal slope.

Here, we consider binary image classification for grayscale images[1]. An image can be thought of as a matrix $I \in \mathbb{R}^{H\times W}$, where each entry contains a

[1]Everything generalizes from grayscale to the RGB case; typically you would aggregate the gradient values over the three color channels by taking the mean. Everything also generalizes to multi-class classifiers $f(I) \in [0,1]^c$ for c classes by including an output channel per class and considering the class of interest.

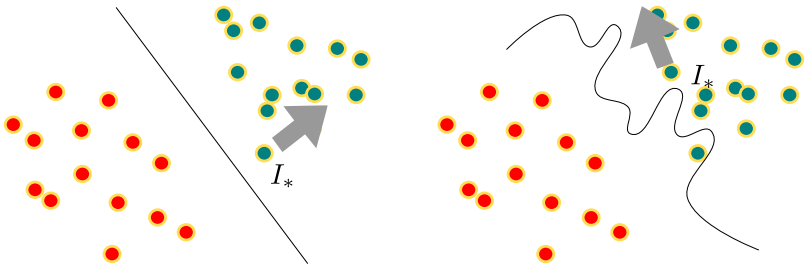

Figure 1: For linear classifiers (left), the gradient consists of the feature weights, which are frequently used for interpretation. It also encodes the direction (in image space) of maximal increase in prediction confidence. That last interpretation also generalizes to nonlinear classifiers (right), where the gradient tells you (locally, with a per pixel velocity) how to move away from the classification boundary as efficiently as possible.

grayscale value. In this case, an image classifier is a mapping $f\colon \mathbb{R}^{H\times W} \to \mathbb{R}$ predicting probability, or confidence, of belonging to the positive class. Now the gradient $\nabla_I f(I_*)$ of $f\colon \mathbb{R}^{H\times W} \to \mathbb{R}$ describes the direction from I_* in $\mathbb{R}^{H\times W}$ in which the slope of f is maximal. When f produces the probability, or confidence, of belonging to the positive class, and $\mathbb{R}^{H\times W}$ is the space of $H \times W$ images, the gradient also belongs to $\mathbb{R}^{H\times W}$ and can be visualized as an image – the saliency map. This gradient encodes the infinitesimal change to the image which maximally increases the predicted probability, or confidence, of belonging to the positive class. Close to the classification boundary, the gradient will point perpendicularly away from the classification boundary, as illustrated in Fig. **??**.

For an image classification problem, this corresponds to answering the question:

> **Fundamental Question 2** *Which pixel intensities should I change at which rate in order to most efficiently get a more certain positive prediction?*

Gradients are quite general – in order to compute gradients, the classifier f needs to be a differentiable predictive model with fixed parameters, such as a trained neural network. The raw gradient $\nabla_I f$ of the classifier f with respect to the image I is typically referred to as the **vanilla gradient saliency map**.

Smoothgrad. As the vanilla gradient is literally just the gradient of the classifier, its stability is directly related to the stability of the classifier itself, and vanilla saliency maps tend to look noisy, as illustrated in Fig. 2. As illustrated in Fig. 3 from [8], the gradients fluctuate heavily –

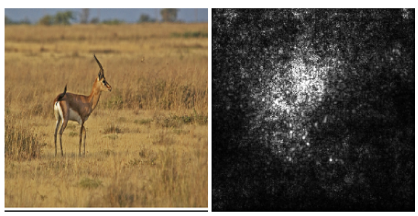

Figure 2: Illustration from [8]: Noisy vanilla saliency maps.

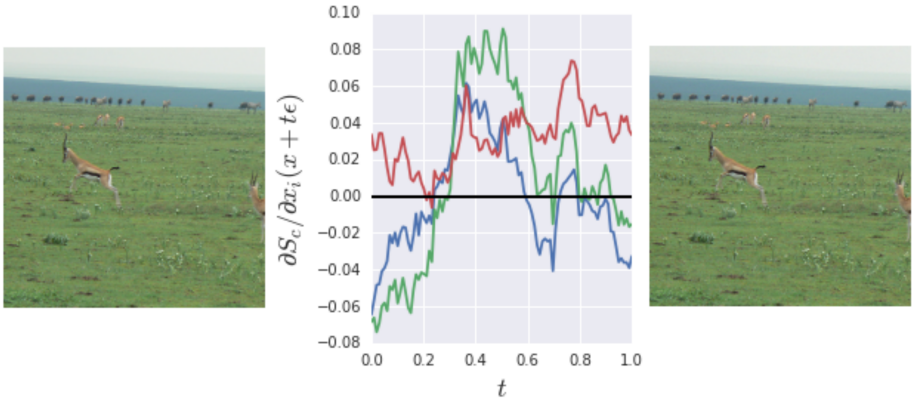

Figure 3: Illustration from [8] showing how the image gradient varies when viewed from a single pixel's red, green and blue channels, interpolating between an image I_* and its small perturbation $I_* + \epsilon$.

they even change signs, indicating opposite effects on classification – as one interpolates between an image and a small perturbation of it.

The **smoothgrad** [8] algorithm offers a simple remedy: Apply Gaussian smoothing of the saliency map over image space to obtained a smoothened saliency map $\hat{S}(I_*)$. In practice, such a Gaussian smoothing is carried out via stochastic approximation:

$$\hat{S}(I_*) = \frac{1}{n} \sum_n \nabla_I f(I_n), \tag{1}$$

where I_n is drawn randomly from the normal distribution $\mathcal{N}(I_*, \Sigma)$. Typically, the covariance matrix Σ is of the form

$$\Sigma = \begin{pmatrix} \sigma^2 & 0 & \dots & 0 \\ 0 & \sigma^2 & \dots & 0 \\ & & \ddots & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & \sigma^2 \end{pmatrix} \in \mathbb{R}^{(H\times W)\times(H\times W)}, \tag{2}$$

encoding covariance between different pixels on the $H \times W$ pixel grid.

Remark 1 *This form of the covariance matrix makes it particularly easy to sample the images I_n!*

Note the form of the covariance matrix Σ in Eq. (2). Its diagonal form indicates that the different pixels vary independently, and the constant σ^2 entries along the diagonal indicate that every pixel follows a normal distribution with variance σ^2. Thus, the sampled image I_n from Eq. (1) can be obtained from

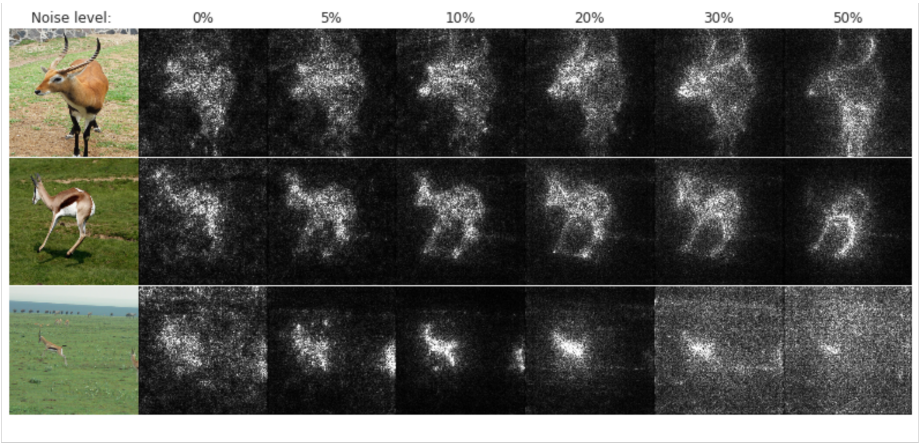

Figure 4: Illustration from [8] showing how the result of smoothgrad varies for different noise levels, where the noise level is given by $\sigma/(I_{max} - I_{min})$.

I_* by adding independently sampled Gaussian noise with variance σ^2 to every pixel.

As one might expect, the result of applying smoothgrad depends on the level of smoothing encoded in the variance σ^2, which becomes a hyperparameter to be tuned on its own. This effect is shown visually in Fig. 4.

Integrated gradients. An alternative way to handle the robustness issue is given by the Integrated gradients algorithm [9]. The integrated gradients algorithm computes a saliency map using Eq. (1), but where samples are drawn as follows: Select a *baseline image* B – the default is a zero image (a black image), but the baseline could be considered a hyperparameter in itself. Draw image space samples I_n spaced equally along the linear interpolation between the images B and I_*.

Performance will depend on the choice of baseline image B; example results are found in Fig. 5.

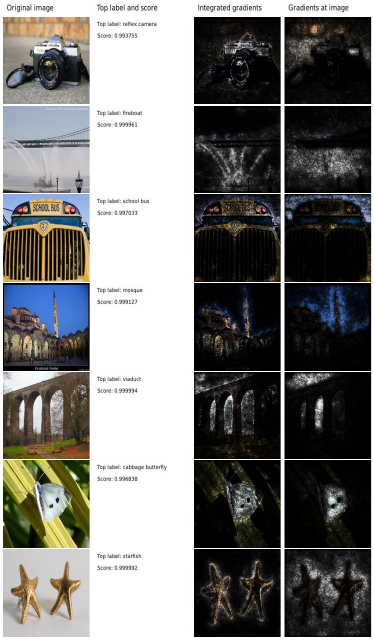

Figure 5: Example results of Integrated gradients from [9].

Context: Gradients as explanations. Using gradients to create explanations for classifiers is well established. Gradient-based explanations can be thought of as

generalizing the inherent interpretability of linear models: The gradient of a linear regression model is precisely its weights, which are often used to interpret feature importance. Similarly, the weights of a logistic regression model form the gradient of the logit function preceding the final sigmoid. As shown in Fig. 1 (left), the gradient of a linear classifier such as logistic regression points orthogonally away from the classification boundary. As such, it encodes very explicitly how one should change the features of an input in order to, as efficiently as possible, move towards a more extreme positive classification. Note, however, that this gradient is completely independent of the input image – it describes the model alone, and would generate the exact same saliency map for any input image. This insight incidentally *also* illustrates why linear classifiers are not good predictors for raw images: They expect to see the same discriminative object in the same coordinate every time.

Conversely, Fig. 1 (right) illustrates a nonlinear image classifier, whose classification boundary is curved and where the landscape defined by the predictive probability of the positive class is highly nonlinear. Here, the gradient varies with spatial location. In other words, it depends closely on both the image of interest and on the model, and tailors the explanation to the classifier's behavior for this particular input image. So far, so good. However, Fig. **??** (right) also shows that even if there is only one gradient, there can be many quick routes to the classification boundary. This is a first indication that our fundamental question may be ill posed – a point to which we shall return.

2.2 Saliency maps based on activation mappings

An alternative, popular way to define saliency maps, is via **activation mappings**. This originates with the *Class Activation Mappings* (CAM) method [11], which applied to classification networks where the layer preceding the final predictive layer was constrained to be a channel-wise global average pooling, see Fig. 6. To obtain a saliency map, the activation maps from each channel A^k in the second last layer were aggregated as a weighted sum, where the weights w_k were defined by channel-wise global average pooling in the penultimate layer:

$$S(I_*) = \sum_k w_k A^k.$$

The resulting image was then upsampled to the resolution of the original image.

The need for a global average pooling is not all that attractive, and the GradCAM method [7] alleviates this by adapting CAM to handle any model in a post-hoc fashion. Here, the activation mapping weights are instead computed via gradients:

$$w_k = \text{mean}(\nabla_{A^k} f(I_*)$$

where $f(I_*)$ is the (before softmax) prediction for input image I_*, and A^k is the features of the k^{th} penultimate channel, see Fig. 7. Now, the GradCAM saliency map is defined as

$$\hat{S}(I_*) = ReLU(\sum_k w_k A^k),$$

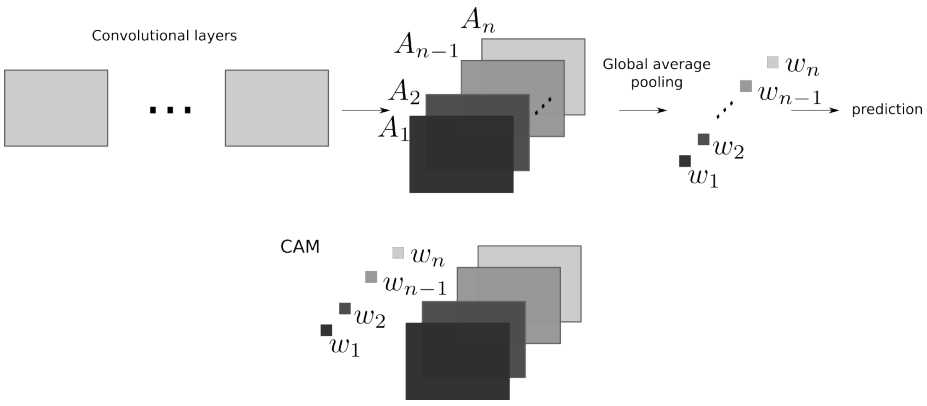

Figure 6: For the original CAM method, the architecture was constrained in the sense that the final layer prior to prediction had to be a channel-wise global average pooling. To obtain a saliency map, the activation maps from the second last layer were aggregated as a weighted sum, where the weights were defined by the global average pooling layer. The resulting image was then upsampled to the resolution of the original image.

Figure 7: GradCAM improves on CAM by using channel-wise gradients to compute activation mapping weights, instead of requiring a global average pooling layer.

followed by upsampling to the original resolution of the image. Note the use of ReLU, which turns off negative contributions – this leads to a more intuitive output.

While the descriptions above assume binary classification models, the descriptions carry over directly to multi-class classification by applying the definitions to the class of interest. Fig. 8 shows how applying GradCAM to two different classes – a species of cat and a species of dog – give different saliency maps emphasizing different parts of the image.

2.3 Attention-based saliency maps.

A final, more modern, way to create saliency maps, is by visualizing attention maps embedded in the model itself [5]. These attention maps are crucial elements of the transformer architecture, but they are also commonly used in CNNs, where they refer to a trained weight image which is multiplied onto a feature image. An example is found in Fig. 9.

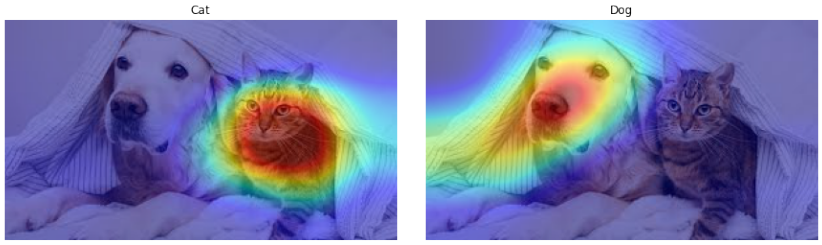

Figure 8: GradCAM applied to two different output channels of the same mode, corresponding to a species of cat and a specias of dog. Note how the resulting saliency maps highlight different regions in the image.

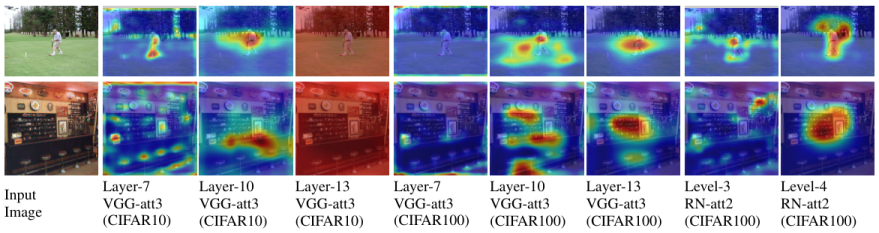

Figure 9: Attention maps appearing as an integrated part of transformers or CNNs can also be upsampled and visualized as saliency maps. Figure from [5].

3 Adversarial attacks

A concept closely related to gradient-based saliency maps, is *adversarial attacks* [10].

Assume given and image $I_* \in \mathbb{R}^{H\times W}$, a multi-class classifier $f\colon \mathbb{R}^{H\times W} \to \{0,\dots,L\}$, and a target class label l. An adversarial attack on the image I_* towards the class l is defined as finding image "noise" $r \in \mathbb{R}^{H\times W}$ that solves the task:

$$\text{minimize} \|r\|^2 \text{ while } \begin{array}{l} f(I_*+r) = l, \\ I_*+r \in [0,1]^{H\times W}. \end{array}$$

This process is illustrated in Fig. 10: We seek a "noise image" r whose magnitude is as small as possible – so that by adding the noise r to the image I_*, the resulting object I_*+r is still a valid image,

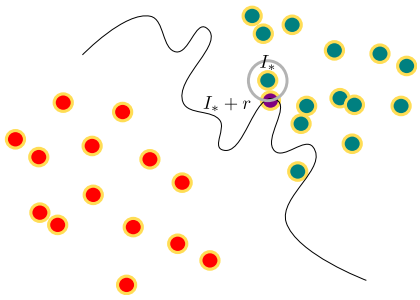

Figure 10: Adversarial attacks aim at finding the minimal noise r perturbing the image I_* so that I_*+r belongs to another class than I_* while looking visually indistinguishable from I_*.

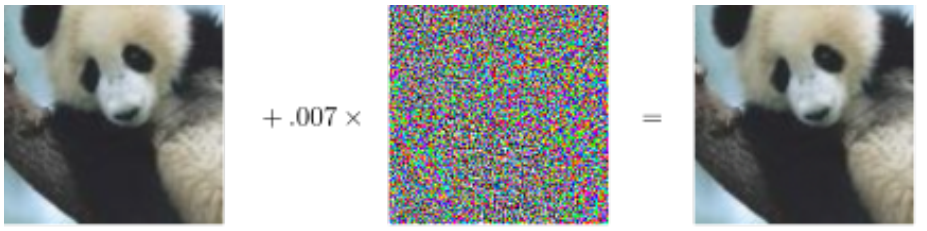

Figure 12: An example from [3] of the FSGM method: By adding a small magnitude "noise" image to the original image I_*, the class of the resulting image $I_* + r$ is changed whereas the images look visually interchangeable.

whose classification by f has changed to the class l. In other words, we want the closest possible crossing of the classification boundary, with the constraint of not moving out of the realm of valid images.

Does this look familiar? Recall – from our discussion of gradient-based saliency maps – that the most efficient way towards or away from the classificaiton boundary, is given by the gradient of the classifier. This realization lies behind two well-known algorithms for creating adversarial attacks.

The fast gradient sign method (FGSM). The FGSM method obtains adversarial examples via "noise" [3] by defining a noise vector r as

$$r = \varepsilon \text{sign}(\nabla_I \mathcal{L}(\theta, I_*, y))$$

where $\mathcal{L}$ is the loss function used to train the classifier, I denotes an image variable, y denotes the ground truth label, ε is a scaling seelected so that the resulting image $I_* + r$ is still an image and the model parameters are denoted θ. This corresponds to moving away from the correct class by maximizing its loss (a linear approximation of the gradient). The components of the approximated gradient are binarized for robustness and modulated with a scaling parameter ε to ensure that the resulting attack image $I_* + r$ is still a valid image.

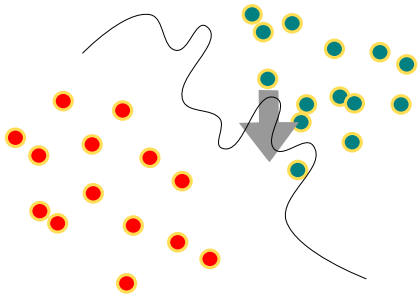

Figure 11: The FSGM method takes a step in the approximate direction of the gradient of the predictor.

The Basic Iterative Method (BIM). The BIM method [6] for creating saliency maps takes a more global view, moving from the original image to its adversarial attack not in a single gradient step, but rather in a series of steps that reflect the local changes in gradient direction of the loss function.

The BIM adversarial attack on an image I_* is defined as:

$$I_0^{adv} = I_*,$$

and

$$I_{n+1}^{adv} = \text{Clip}(I_n^{adv} + \alpha \text{sign}(\nabla_I \mathcal{L}(\theta, I, y))),$$

where Clip cuts off image values that go outside $[0, 255]$, and α is a step size (e.g. 1).

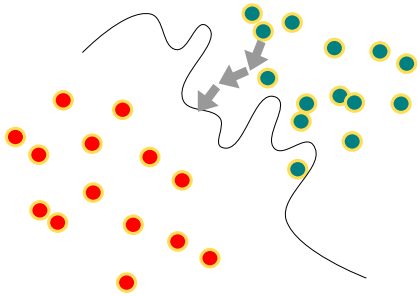

Figure 13: The BIM method approaches the classification boundary in an iterative manner, leading to a more efficient road to the target class when the starting image is far from the classification boundary.

4 The XAI debate – saliency maps as hate objects and whether they deserve it

As artificial intelligence (AI) continues to be implemented into increasingly sensitive aspects of life, the demand to explain AI decisions increases accordingly. As a result, explainable AI (XAI) has become a hot topic of research, where both novel solutions and criticism of these novel solutions get massive attention. Current narratives range from exceedingly optimistic statements such as *"Explainable AI opens the deep learning black box"*, to the very disappointed end of the spectrum: *"[Explainable AI] represents a false hope for explainable AI and [] current explainability methods are unlikely to achieve these goals for patient-level decision support."*

Following the interpretation of saliency maps given in Fundamental Question 1, saliency maps have been criticized widely for not giving robust, intuitive, or even truthful, answers to this question.

In the following, we will first discuss how many popular approaches to creating saliency maps naturally do not provide an answer to this question, and that in fact, the question is ill posed. We will revisit some popular examples of "saliency map failures" and discuss, using nothing but basic calculus, how the "failures" are actually not failures, but perfectly natural behavior of the given saliency map models.

Finally, taking a higher level perspective, we argue that the main problem within explainable AI is not within the XAI models themselves, but rather in the user's understanding and expectations of what those models can and cannot provide. We conclude that education of XAI users is crucial to safe use of XAI.

4.1 Gradient misbehavior: Popular saliency map failures, and how to understand them

Gradient-based saliency maps are often showcased as examples of un-intuitive or even incorrect explanations supplied by XAI. Here, we discuss some common examples and explain why, by understanding what the gradient actually encodes, these examples are neither surprising nor particularly disappointing – the real source of disappointment is the user's incorrect expectations of what the gradient should do.

Highlighting contextual information. A common criticism of saliency maps come from examples where the saliency map highlights regions of the image that we think should be irrelevant to the classification problem. In Fig. 14 we see an example from [1], which studies a medical imaging algorithm that predicts breast cancer diagnoses from screening mammograms. The authors of [1] argue that the algorithm should *only* be looking inside the cancerous lesions, and that the saliency maps whose results are shown in Fig. 14 are therefore failing at their task.

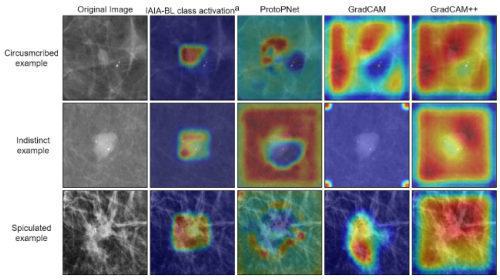

Figure 14: Figure from [1] comparing different types of saliency maps to the method IAIA-BL suggested in [1].

We argue, however, that if the algorithm is actually *using* the contextual information, then the XAI feedback should also reflect this. Thus, both the criticism and the validation of XAI based on whether returned explanations match what we *think* the algorithm should be using, is flawed.

Failure to explain adversarial attacks A second point of criticism directed at saliency maps is their failure to explain the change of classification embedded in an adversarial attack. Fig. 15 shows how several different types of adversarial attacks – including the two discussed above – fail to change a series of different types of saliency maps – also including several discussed above. This is often discussed as a failore of the saliency map [4, 2]. However, in light of our descriptions of, in particular, gradient based saliency maps and adversarial attacks above, this behavior is perfectly natural. Most saliency maps embed, in one way or another, the gradient of the predictor. The adversarial attack consists of taking as small steps as we can get away with, more or less in the direction of these gradients. Knowing these two facts, you would only expect the saliency maps to change dramatically after an adversarial attack if the gradient of the predictor changes dramatically as you take small steps in its direction – a behavior that we otherwise go to great lengths to avoiding in neural networks, using regularization, large datasets, etc. We can therefore conclude that the "failure"

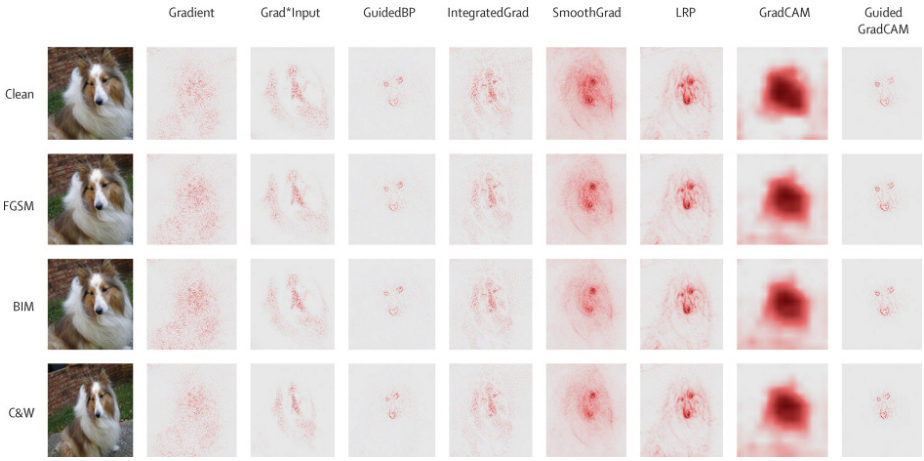

Figure 15: An example from [4, 2] showing how saliency maps are unaffected by an adversarial attack.

of saliency maps to detect adversarial attacks is not a failure of the method – if anything, it's a failure to communicate what the method actually shows.

5 Where do we go from here? Explainable AI cannot stand without an explanation

In light of the above discussion, it might seem tempting to dismiss gradient-based saliency maps as an example of XAI not being good enough. But this would be a missed opportunity. Gradient-based saliency maps are easy to criticize because they are (fairly) easy to understand. This is their strength – not their weakness.

Explain the explainable AI. A far more valuable lesson would be to reinstate the need for practitioners to understand the models they are using – also when those models seek to explain the decisions made by another AI model. We need to educate users of XAI on what they should and should not expect from the XAI if we want to ensure safe technology usage. This is also crucial for ensuring technology acceptance: If the users expect more from XAI then they can realistically get, they are bound to be disappointed and may in the end choose not to use the XAI – not even for the purposes for which it is actually useful.

Incorrect interpretation of XAI. The number of different models producing saliency maps already indicates that the usual interpretation of saliency maps providing an answer to Fundamental Question 1 is incorrect: A wealth

of different models producing different explanations cannot all answer the same question – or can they? At the very least, our new understanding helps us see how the question answered by saliency maps can be rather different from the question asked. If your image I_0 sits in the middle of the positive class, gradient based saliency maps actually tell you how to move, as efficiently as possible, out of distribution.

The fundamental question of XAI is ill defined. From here on, we rephrase the fundamental question of explainable AI in more general terms as:

> **Fundamental Question 3** *How much does each input feature contribute to a given AI decision?*

We argue, supported by the discussion above, that this question is ill defined. First, while a generic input data point and generic trained neural network will have a unique gradient associated to them, the direction encoded by this gradient is not the only (small) change to the input feature that could lead to a more extreme classification. Second, there are multiple notions of scale involved, both in magnitude of pixel-wise contribution and in the number of pixels making the same contribution – if a bird's beak has higher saliency than the bird's wings, the wings might still impact the classification more because they contain far more pixels.

Where do we go from here? For the time being, this leaves you – as technical developers – with a responsibility: If you deploy AI based tools with built-in explainability, you need to make sure that the tool equips the user to actually understand the returned explanations. This can be done via user interfaces, or via teaching – but as the current XAI literature shows us, it does need to be done.

References

[1] Alina Jade Barnett, Fides Regina Schwartz, Chaofan Tao, Chaofan Chen, Yinhao Ren, Joseph Y Lo, and Cynthia Rudin. A case-based interpretable deep learning model for classification of mass lesions in digital mammography. *Nature Machine Intelligence*, 3(12):1061–1070, 2021.

[2] Marzyeh Ghassemi, Luke Oakden-Rayner, and Andrew L Beam. The false hope of current approaches to explainable artificial intelligence in health care. *The Lancet Digital Health*, 3(11):e745–e750, 2021.

[3] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[4] Jindong Gu and Volker Tresp. Saliency methods for explaining adversarial attacks. *arXiv preprint arXiv:1908.08413*, 2019.

[5] Saumya Jetley, Nicholas A Lord, Namhoon Lee, and Philip HS Torr. Learn to pay attention. In *International Conference on Learning Representations*.

[6] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

[7] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

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

[9] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[10] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[11] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.