Neural Computing and Applications (2023) 35:10011–10029
https://doi.org/10.1007/s00521-022-07918-7

S.I.: INTERPRETATION OF DEEP LEARNING

Constructing adversarial examples to investigate the plausibility of explanations in deep audio and image classifiers

Katharina Hoedt[1] · Verena Praher[1] · Arthur Flexer[1] · Gerhard Widmer[1,2]

Received: 1 April 2022 / Accepted: 30 September 2022 / Published online: 26 October 2022

Abstract
Given the rise of deep learning and its inherent black-box nature, the desire to interpret these systems and explain their behaviour became increasingly more prominent. The main idea of so-called explainers is to identify which features of particular samples have the most influence on a classifier's prediction, and present them as explanations. Evaluating explainers, however, is difficult, due to reasons such as a lack of ground truth. In this work, we construct adversarial examples to check the plausibility of explanations, perturbing input deliberately to change a classifier's prediction. This allows us to investigate whether explainers are able to detect these perturbed regions as the parts of an input that strongly influence a particular classification. Our results from the audio and image domain suggest that the investigated explainers often fail to identify the input regions most relevant for a prediction; hence, it remains questionable whether explanations are useful or potentially misleading.

Keywords Interpretability · Explainability · Adversarial examples · Evaluation

1 Introduction

In recent years, a wide variety of explanation methods ("explainers") have been developed in order to explain the inner workings of deep neural networks, with two large groups being perturbation- and gradient-based explainers. The large number of approaches available makes it hard or even impossible to decide which explanation technique to choose and which explanation to trust. Since there is no agreement on what constitutes a "good explanation" [1], an explanation is often simply evaluated based on whether it looks reasonable or matches one's expectation ("confirmation bias") [2].

This lack of standard evaluation metrics hinders a comparison between different explanation techniques. Even worse, it was shown that different explanation methods disagree [3, 4], which suggests that at least some of them do not capture the real inner workings of a system and that explanations produced by an explainer A cannot serve as a ground truth for a newly developed explainer B.

While no standard evaluation procedure for explanations has emerged so far, a few different approaches were proposed in the past. On one hand, these include metrics that quantify the quality of single explanations, e.g. fidelity, consistency, or comprehensibility [5]. These metrics may sometimes be hard to compute (e.g. comprehensibility) and can—in the worst case—give a false sense of trust into a particular explanation (cf. [6]). On the other hand, also approaches for evaluating an explanation algorithm as a whole were proposed, e.g. region perturbation [7], ROAR [8], sanity checks [2], and the CLEVR-XAI benchmark [9]. But again, the assumptions underlying many of these approaches have been questioned in the past (cf. [9, 10]).

Previous approaches to compare different explainers either focus on pixel-wise attributions computed by gradient-based explainers (without considering perturbation-based explainers like Local Interpretable Model-agnostic Explanations (LIME)) [2, 8, 9] or compare LIME to rule-

Katharina Hoedt and Verena Praher have contributed equally to this work.

✉ Katharina Hoedt
katharina.hoedt@jku.at

✉ Verena Praher
verena.praher@jku.at

[1] Institute of Computational Perception, JKU Linz, Linz, Austria

[2] LIT AI Laboratory, Linz Institute of Technology, Linz, Austria

based techniques or other methods, which are often not applicable to domains such as images [11, 12]. A comparison between gradient-based and perturbation-based explanations is not straightforward, as the granularity of these explanations is different; as opposed to pixel-wise explanations, LIME explanations consist of groups of pixels, also called super-pixels or segments. Recent work has therefore tried to address evaluation from a different angle, namely by using dataset modifications [13] or adversarial examples [14] as ground truth. As adversarial perturbations lead to changes in a system's original prediction, the reason for such an (erroneous) prediction can be found in these perturbations of the input. Göpfert et al. [14] use so-called localised adversarial attacks, which attack a constrained region of an input image, in order to evaluate classic Saliency [15], Guided Backpropagation [16], and LIME [17], making it one of the first attempts to compare pixel-wise attribution maps with a method such as LIME. The evaluation in Göpfert et al. [14] is based on how well each explainer recovers the affected region as the cause for the adversarial prediction, with LIME being the explainer that performs best in their setting.

In this work, we follow a similar idea and use adversarial perturbations to evaluate the plausibility of explanations in two different domains, namely audio as well as images. We refine and extend the analysis along several dimensions, in order to provide more detailed insights and a fairer comparison for different explainers. In particular, we test explainers using adversarial perturbations spread over either parts or the entirety of a sample. Since we are not restricting perturbations to very dense regions of a sample, we do not favour explanations which are inherently more concentrated (such as LIME).

In a bit more detail, the paper is structured as follows: First, we give an overview over all examined explainers in Sect. 2, as well as the attacks on the audio / image systems in Sect. 3. We then detail our experimental setup in Sect. 4. In Sect. 5, we evaluate LIME on an audio classification task for which we chose a state-of-the-art Singing Voice Detection System (SVDS) [18]. In this set of experiments, we focus on LIME with time-frequency segments (i.e. rectangular subparts of spectrograms) as an explainer, as this is the most prominent method in audio (cf. [19–22]). Then, in Sect. 6, we evaluate a wider range of explainers, including earlier approaches (e.g. LIME, Saliency and Guided Backpropagation), as previously compared by Göpfert et al., as well as more recent methods (e.g. SmoothGrad [23] or VarGrad [2]) on a variety of pre-trained ImageNet classifiers. For the experiments in the image domain, we use more commonly used super-pixels computed by the Simple Linear Iterative Clustering (SLIC) algorithm [24] as segments, as well as rectangular image segments. We also propose to aggregate pixel-wise attributions to such super-pixels or segments. In contrast with Göpfert et al. [14], we propose a non-random baseline based on the magnitude of the adversarial perturbation for all our experiments. We finally discuss our findings and give an outlook in Sect. 7. This article is an expanded version of a conference paper (Praher et al. [6]), with the following new contributions:

- We present the first work that compares different explainers for multiple models in the image domain as well as an explainer applied to an audio system, using adversarial perturbations as ground truth;
- We propose to aggregate pixel-wise attributions computed by gradient-based explainers for an improved comparison with LIME;
- Finally, in contrast with [14], we compare explanations with a stronger, non-random baseline.

2 Explainers

This section contains a brief description of a variety of methods that are often used to explain the decisions of computational models, all of which we subsequently compare in different experiments. Each of these following approaches is *local* (explains individual predictions) and *post hoc* (can be used on a trained model without modifying it). An explainer can be model-specific (it only works for a subset of model classes with specific properties, e.g. gradient-based methods) or model-agnostic (e.g. LIME). Model-agnostic explainers can be applied to any model type and only require inputs and the corresponding outputs [25]. For more detailed descriptions, we refer the reader to the original papers.

2.1 Local interpretable model-agnostic explanations

The main idea of LIME [17] is to approximate the neighbourhood of a prediction $f(x)$, which we want to explain, with a model g that is simpler than f (most commonly, g is linear). To do this, the first step requires the derivation of an interpretable representation depending on the input domain. For images, this can be (perceptually) grouped pixels, also called super-pixels.

Let $x \in \mathbb{R}^d$ be a sample in the original input domain[1], and let $x' \in \{0,1\}^{d'}$ be the interpretable representation (0 / 1 denote the absence / presence of interpretable features). We then sample N_s instances around x' by randomly generating vectors containing 0's or 1's. Each of these

[1] E.g. $d = \text{width} \times \text{height} \times \text{channels}$ for images, or $d = \text{time} \times \text{frequency}$ for input spectrograms.

generated instances z' is subsequently mapped back to $z \in \mathbb{R}^d$, i.e. its representation in the input domain; z can here be imagined to look like the original input x, yet with all parts that equal 0 in z' being occluded. All created instances z' and their predictions of the original model $f(z)$ are then used to train the explanation model g [17]. A model g most often corresponds to a linear model, e.g.

$$g(z') = b + w_g z'. \tag{1}$$

Note here that samples z are usually additionally weighted based on how close they are to the original input x during training of the model g.

The most typical interpretable representations used in the image domain are super-pixels (i.e. segments) computed with the SLIC [24] algorithm or any standard image segmentation algorithm [17]. One could also use a neural network to predict image segments. In the audio domain, rectangular time–frequency segments were previously proposed [19].

In what follows, we use these rectangular segments when investigating an SVDS; we also adapt this to our experiments in the image domain, using equally sized rectangular image segments. Additionally, we repeat all experiments in the image domain with SLIC segments.

2.2 Gradient-based explanations

Next to LIME, we want to look at a different and popular family of explainers, based on gradients. To avoid confusion between the general term saliency maps (often used to refer to the attribution maps that serve as explanations) and the approach "Saliency Maps", we are going to use "Saliency Maps" to refer to the concrete algorithm and "attribution maps" when we talk about the resulting visualisation.

Saliency Maps (Sal) [15] (also called "Classic Saliency" [14] or "Gradient" [2, 8]) are one of the first approaches for visualising the importance of each pixel for a class decision ("which pixels need to be changed the least to affect the class score the most"). This is achieved by taking the derivative of the class score S_c with respect to the input image x, i.e.

$$A = S_c'(x). \tag{2}$$

Guided Backpropagation (GBP) also takes the derivative of the class score S_c with respect to the input image, but additionally overrides the backpropagation of the ReLU function in a way to only backpropagate non-negative gradients. This leads to sharper visualisations than previous methods [16].

Integrated Gradients (IG) [26] addresses different weaknesses of previously published methods. It computes the straight-line path from a baseline x^0 (e.g. a black image) to the input of interest x and accumulates all gradients at k different points along that path, i.e.

$$A = (x - x^0) \times \sum_{i=1}^{k} \frac{\partial f(x^0 + \frac{i}{k}(x - x^0))}{\partial x} \times \frac{1}{k}. \tag{3}$$

Note that f here denotes the model we want to interpret.

2.3 Smoothing noisy attributions

Attribution maps like the ones in Sect. 2.2 often look "noisy" to a human observer [26] as shown in Fig. 1. In the following, we therefore describe different approaches for smoothing attribution maps. Smoothing can be applied to any of the gradient-based explainers, resulting in 9 different combinations (subsequently indicated by a concatenation of their abbreviations, e.g. IG-SG for Integrated Gradients with SmoothGrad). Examples are shown in Appendix in Fig. 9. Following the terminology in [8], we will refer to the smoothed versions of the previously explained standard explainers as "ensemble explainers".

SmoothGrad (SG) [23] averages a set of J noisy attribution maps, which are constructed by adding Gaussian noise $\mathcal{N}(0, \sigma^2)$ with standard deviation σ to the input of interest x independently J times:

$$A = \mathbb{E}_{\varepsilon_j \sim \mathcal{N}(0,\sigma^2)}\left[S_c'(x + \varepsilon_j)\right]. \tag{4}$$

SmoothGrad-Squared (SG^2) is based on SmoothGrad with the small difference that the each of the J attribution maps is squared before averaging [8], i.e.

$$A = \mathbb{E}_{\varepsilon_j \sim \mathcal{N}(0,\sigma^2)}\left[S_c'(x + \varepsilon_j)^2\right]. \tag{5}$$

VarGrad (VG) is also based on SmoothGrad, but instead of averaging, the variance over J noisy attribution maps is computed [2], i.e.

$$A = \mathrm{Var}_{\varepsilon_j \sim \mathcal{N}(0,\sigma^2)}\left[S_c'(x + \varepsilon_j)\right]. \tag{6}$$

2.4 Hyper-parameters for the explanations

LIME: The hyper-parameters for LIME were chosen based on preliminary experiments. For audio explanations, we occlude segments based on their *mean* spectrogram value (cf. [22]). We set the number of perturbed samples to $N_s = 8,192$ for explanations in the audio domain. This value is set to $N_s = 512$ in the image domain as a trade-off between stability [22] and run time. We use an exponential kernel defined on the cosine distance function with kernel width 0.25 for explanations in both domains.

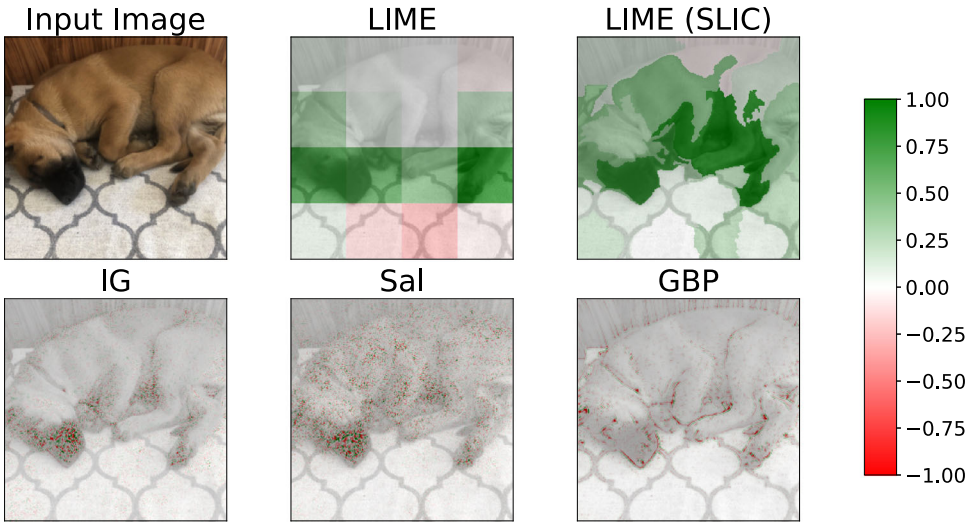


Fig. 1 Example explanations for each of the used standard explainers. Different colours visualise the weights/attributions that different explainers assign to particular segments/pixels, with higher numbers indicating more, and lower numbers less "important" features for a prediction

GBP, Sal, IG: Gradient-based explainers have less hyper-parameters that need to be considered. GBP and Sal do not have any hyper-parameters. For IG, we need to select the baseline, for which we choose a black image (all zeroes).

SG, SG², *VG:* For SG, SG² and VG, we set J (the number of noisy attribution maps that are used for smoothing) to 25, as suggested by [8].

3 Adversarial attacks

To attack different image classifiers, we apply a targeted version of Projected Gradient Descent (PGD), originally proposed by Madry et al. [27]. The targeted PGD attack iteratively changes an input image such that its system loss w.r.t. a new target prediction is decreased, i.e.

$$\tilde{x}_{ep+1} = \text{clip}_{\epsilon}(\tilde{x}_{ep} - \eta * \text{sign}\,(\nabla_{\tilde{x}_{ep}} L(f(\tilde{x}_{ep}), t))). \tag{7}$$

Every adversarial example $\tilde{x}$ is subsequently initialised with the original (clean) image x. In each iteration ep, $\tilde{x}$ is updated based on the sign of the gradient w.r.t. the input (i.e. $\nabla_{\tilde{x}}$) of the system loss L. Note that f denotes a system, and t denotes a particular target. The perceptibility of an adversarial perturbation is influenced by the step size η, as well as the clipping factor ϵ, which is used to ensure that a perturbation stays within the range of $[-\epsilon, \epsilon]$.

We use this attack when attacking image classifiers to avoid unnecessary high runtime complexity (cf. exemplary adversaries in Appendix B.2). For audio data, it was previously shown [28] that additional restrictions when using a Carlini & Wagner (C&W) like attack as opposed to PGD leads to less perceptible audio perturbations. We therefore apply C &W to attack the SVDS. This results in a modified adversarial objective (cf. [28, 29]), now expressed as an optimisation for the adversarial perturbation δ (i.e. $\tilde{x} = x + \delta$) itself:

$$\begin{aligned} L_{\text{total}} &= \|\delta_{ep}\|_2^2 + \alpha * L_{\text{sys}}(f(x + \delta_{ep}), t), \\ \delta_{ep+1} &= \text{clip}_{\epsilon}(\delta_{ep} - \eta * \text{sign}\,(\nabla_{\delta_{ep}} L_{\text{total}})). \end{aligned} \tag{8}$$

The notation here remains the same; the most significant change is the integration of the squared L2-norm of the perturbation $\|\delta\|_2^2$ as an attempt to keep it as small (imperceptible) as possible. The factor α balances the focus of the attack on either keeping a perturbation as small as possible, or changing the prediction on the adversary quickly.

To find hyper-parameters for PGD, we perform a grid search over values for the clipping factor ϵ and the update rate η on a subset of our image data. We choose the hyper-parameters based on what leads to the highest number of successful attacks with the overall smallest difference in magnitude between clean and adversarial samples. The maximal number of iterations we use to search for a successful adversarial example is 100; the target t is chosen for each sample separately to be any random but different prediction (in comparison to the prediction on a clean sample). For C&W, we set the maximum number of iterations to 1000; the remaining hyper-parameters, i.e. clipping factor ϵ, update factor η and weight factor α, are tuned on the audio validation set (cf. Sect. 4.1). They are chosen as the setting that leads to the highest number of successful adversarial perturbations. The target t for each sample is, due to the binary nature of the SVDS, the class that is not the original prediction. For exact hyper-parameters, we refer to Tables 4 and 5 in Appendix.

4 Experimental setup

In this section, we describe the experimental setup, including the data we use and the classifiers we investigate in our experiments. We provide the code via Github[2]; for further implementation details, we refer to Appendix C.

4.1 Data

To train the singing voice detection system (cf. [6]), we use the openly available *Jamendo* dataset [30], which consists of 93 songs resulting in roughly 6 hours of music. The training / validation / test split is proposed to contain 61 / 16 / 16 songs, respectively, with roughly the same proportion of annotated singing voice versus non-singing voice in all three splits. The audio files have a sampling rate of 44.1kHz, and are annotated with the presence ("sing") or absence ("no sing") of singing voice on a sub-second granularity [31]. Note that during training and test time, songs are not used as a whole, but instead split into a multitude of smaller excerpts (length $\approx$ 1.6 seconds).

To extend these experiments to the image domain, we use the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset [32]. The dataset contains 1,000 different classes and provides around 1.2 million images for training, as well as 50,000 images for validation (i.e. 50 per class). Subsequent experiments are performed on all validation images or a subset thereof[3]. Hyper-parameters are tuned on validation data as well, as we assume an attacker as well as people looking for explanations have access to the respective data.

4.2 Singing voice detection system

For experiments on audio data, we use the SVDS previously introduced by Schlüter and Lehner [18]. Data preprocessing, which consists of resampling and computing normalised magnitude Mel spectrograms, as well as the training procedure itself is done as proposed for the suggested Convolutional Neural Network (CNN) architecture.

To compute performance metrics of the SVDS, we use binary predictions of the network, which are obtained by application of a median filter followed by thresholding (cf. [18]). The classification error (%) on the test data is 11.54 ± 0.96 (given as mean $\pm$ standard deviation over 5 runs with different random initialisations). Recall and specificity are 89.61 ± 1.71 and 87.46 ± 1.00, respectively.

[2] https://github.com/CPJKU/plausible_xai.

[3] Annotated test samples are only available for the challenge itself.

4.3 Image classifiers

To perform experiments on image data, we look at a variety of different pre-trained image classifiers. Other work on evaluating explanations (cf. [2, 8, 14]) investigated Inception v3 [33] and ResNet-50 [34]. In addition to these networks, we compare our results on AlexNet [35], VGG16 [36], and DenseNet161 [37].

The top-1 accuracy of each pre-trained model on the validation data of ImageNet is shown in the second column in Table 1, as well as the accuracy of the classifiers after an attack (column 3). For more information regarding the networks we used, we refer to our implementation details in Appendix (section C).

5 Evaluation of LIME in the audio domain

In our first set of experiments, we look at explanations of the predictions on singing voice data. We first compute adversarial excerpts and explanations for their predicted class. Each of the explanations consists of a list of 20 interpretable features (rectangular segments) and their corresponding weight, which is interpreted as its importance for making a prediction [17, 19]. We also present a baseline for choosing candidates of "most influential" segments of an input.

5.1 Explaining predictions on adversarial examples

To evaluate whether LIME can successfully detect causes for a prediction, we use (imperceptible) adversarial perturbations as ground truth. The goal is to investigate whether the explanation can highlight the segments that are responsible for the adversarial (and new) prediction.

5.1.1 Using a fixed number of interpretable features

It is often claimed that a small number of interpretable features k are sufficient to explain a classifier's prediction [17]. In the particular case of our singing voice detector, a choice for k in existing work is 3 [19].

We use LIME to explain predictions after successful adversarial attacks where perturbations of the original signal could have appeared anywhere in the spectrogram and also on possibly more than 3 segments. We then perform another attack adding only those perturbation segments coinciding with the 3 segments highlighted by LIME and evaluate how well they are able to change a prediction. We argue that if these highlighted segments actually explain a particular prediction, we can expect using this

Table 1 Top-1 accuracies of different image classifiers. The second column shows the accuracy on all 50,000 validation samples; the third column contains the accuracy on 50,000 samples where we predict on an adversary if available, or else the clean image. The final column shows the number of successful adversaries

Architecture	ImageNet (val)	ImageNet (adv)	#Adversarial examples
AlexNet	56.52 %	0.19 %	49,879
VGG16	71.59 %	0.17 %	49,912
ResNet-50	76.13 %	0.16 %	49,925
DenseNet161	77.14 %	0.19 %	49,899
Inception v3	77.21 %	0.38 %	49,846

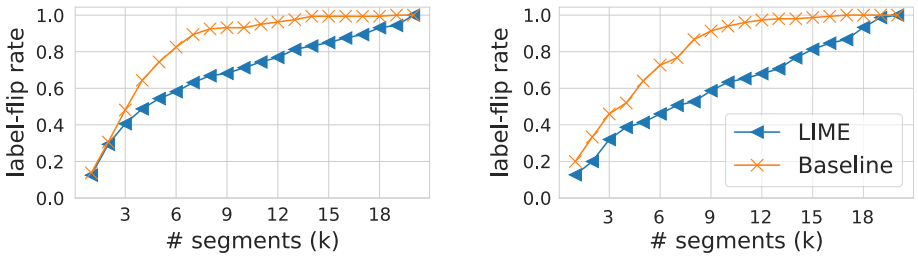


Fig. 2 Label-flip rate (y-axis) for LIME compared to norm-based baseline. The x-axis shows number of perturbed segments that are added to clean input

(a) "no sing" → "sing" **(b)** "sing" → "no sing"

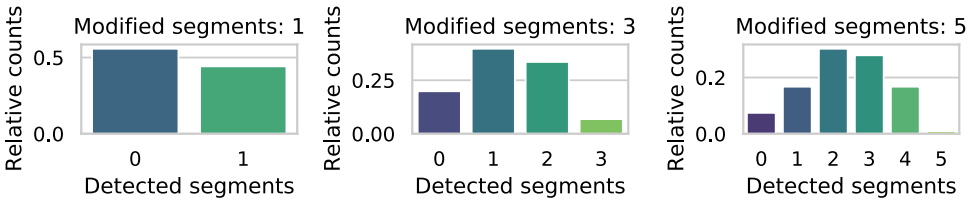


Fig. 3 Relative number of segments correctly recovered by LIME, after adding only k perturbed segments to each sample. We look at 52 / 146 / 215 adversaries in total for $k = 1/3/5$

subset of perturbed segments to achieve the same change in prediction as the full perturbation did. However, using $k = 3$ changes only 32% ("sing" → "no sing") and 41% ("no sing" → "sing") of the predictions. Subsequently, we call this metric the *label-flip rate*.

5.1.2 Varying the number of interpretable features

As fixing $k = 3$ could also simply be a suboptimal choice for k, we perform this analysis for all $k \in \{1, ..., 20\}$. Additionally, we select k perturbation segments based on their magnitude (i.e. the squared L2 norm) as a baseline. The according label-flip rates are shown in Fig. 2.

Almost regardless of the value for k, the explanations provided by LIME do not reach the performance of the baseline. For LIME-chosen segments, it requires almost all 20 segments to achieve label-flip rates close to 100%, whereas for the baseline, this is the case with a fraction of the number of segments.

5.2 Explaining 'localised' perturbations

As it is often sufficient to add even small numbers of selected segments of a perturbation and still achieve new classifications, we conduct an additional experiment on how different explainers can recover changed segments of such *refined* or partial adversaries. We refine adversarial perturbations by splitting them into segments that correspond to the segments we use for LIME (i.e. rectangles). We then choose the k segments with the highest magnitude and use them to perturb the original (clean) data. In what follows, we show results on adversarial subsets of the data, namely samples for which adding $k \in \{1, 3, 5\}$ adversarial segments change "sing" to "no sing" or vice versa.

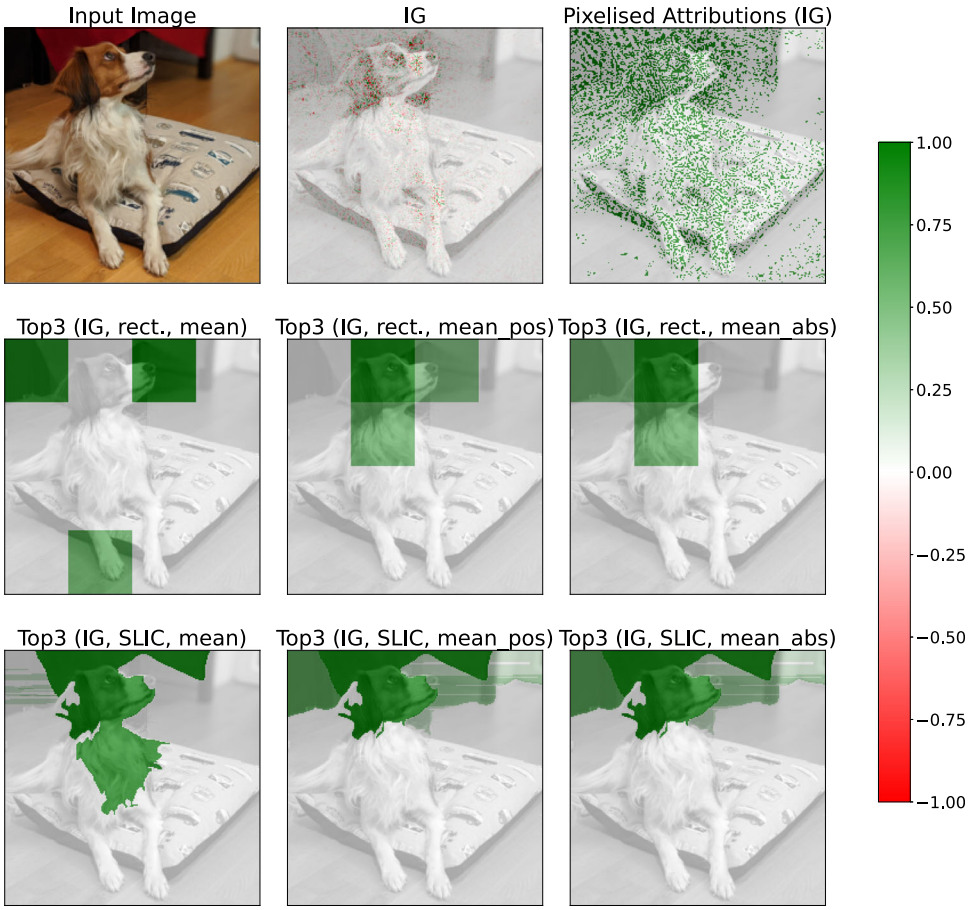


Fig. 4 First row shows the original input image, the raw IG attributions, and the most relevant pixels in binarised form (as used in [14]). The second and third rows show the proposed techniques for aggregating pixel-wise attributions using rectangular and SLIC segments, respectively. Colours again show importance of segments, with higher values denoting higher importance

Table 2 Label-flip rates for standard explainers (SLIC segments, $k = 3$)

Model	LIME	IG	Sal	GBP
AlexNet	68.54	52.16	57.78	60.18
VGG16	55.44	40.90	40.60	49.44
ResNet-50	44.70	27.18	27.10	36.20
DenseNet161	37.82	22.42	22.84	19.42
Inception v3	40.86	28.14	26.52	27.90

We use LIME to compute explanations for this set of refined adversaries and set the number of interpretable features k to be equal to the number of perturbed segments in a sample (i.e. 1, 3 or 5). Since the segments of the explainer and the partial perturbations align, we can examine how often LIME is able to recover the correct segments that are perturbed, and hence are responsible for the new prediction. The result of this experiment is shown in Fig. 3; even for $k = 1$, LIME shows the correct segment as explanation in less than half of the cases. Also for $k = 3$ or $k = 5$, the explainer rarely finds all modified segments,

Table 3 Label-flip rates for ensemble explainers (SLIC segments, $k = 3$)

	Smoothgrad			Smoothgrad^2			Vargrad		
Model	IG	Sal	GBP	IG	Sal	GBP	IG	Sal	GBP
AlexNet	50.68	54.32	49.60	50.70	54.84	47.84	50.34	54.82	47.34
VGG16	39.42	43.32	39.72	40.84	43.30	38.22	41.68	44.36	40.82
ResNet-50	33.26	34.28	33.74	34.06	35.54	33.56	33.82	35.40	32.38
DenseNet161	27.24	28.10	14.10	27.18	29.28	14.62	26.48	29.04	14.52
Inception v3	33.74	35.38	25.22	33.60	36.04	24.18	35.50	36.66	23.46

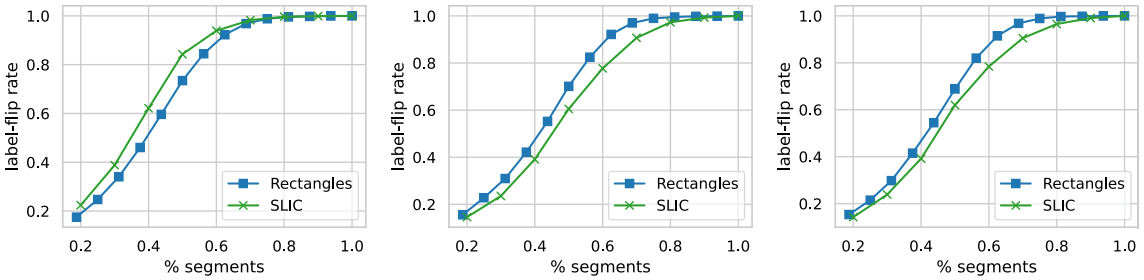


Fig. 5 Label-flip rates of different explainers with rectangular and SLIC segments (ResNet-50) for different percentage of perturbed segments added

and in some settings none of the segments responsible for a prediction.

6 Evaluation of explanations in the image domain

We extend our previous experiments by analysing image classification. We once more split all perturbations into the segments (here: image subparts) that the explainers work with, i.e. now either rectangles[4] or SLIC segments. We are using $4 \times 4 = 16$ rectangular segments or set `n_segments=16` for the SLIC experiments [5]. To ensure a fair comparison between LIME and other explainers, we propose to aggregate pixel-wise attributions using the same segments that LIME uses. The aggregation procedure is explained in more detail subsequently, followed by a set of experiments for evaluating different explanations in the image domain. As a baseline for our explainers in the following experiments, we again use the selection of most important segments as the segments exhibiting the highest norm. Due to the huge computational cost for computing explanations, the experiments in Sects. 6.2 and 6.3 are performed on 10% of the validation dataset.

6.1 Aggregating pixel-wise attributions into segments

In order to compare Sal, GBP and LIME, Göpfert et al. [14] treat the segment weights computed by LIME as pixel-wise attributions. They fix the number of "most influential pixels" that contribute to an explanations for all explainers, and compute the overlap between the pixel-wise attributions and the ground truth, i.e. all pixels in a segment that were adversarially perturbed. This might be unfair toward gradient-based explainers since pixel-wise attributions are often spread out over the whole image yet still could be concentrated around the most relevant regions. The localised perturbations of Göpfert et al. [14], however, only cover constrained parts of the image.

We focus on highlighted *regions* as opposed to highlighted pixels. Even publications about saliency-based attribution maps often talk about finding or highlighting regions [23], highlighting areas or regions [38], or "descriptive image regions" [16] rather than pixels. We further motivate this approach by the fact that attribution maps look like they are highlighting regions, e.g. CAM [39], Grad-CAM [40], Occlusion [41], and Deconvolution [41]. Recently, there have also been attempts at smoothing noisy attribution maps in order to highlight meaningful regions instead of seemingly random pixels [23].

Due to the fact that the goal of many of these explainers is highlighting regions rather than single pixels, it seems fair to also evaluate their capability in doing so. We propose aggregating pixel-wise attributions per segment to receive one importance value per segment, as we obtain for LIME. The only other work we are aware of who aggregate pixel-wise attributions into segments (or "patches") [42] uses an aggregation method called "total patch saliency", without specifying how pixels are actually aggregated.

We compare three approaches:

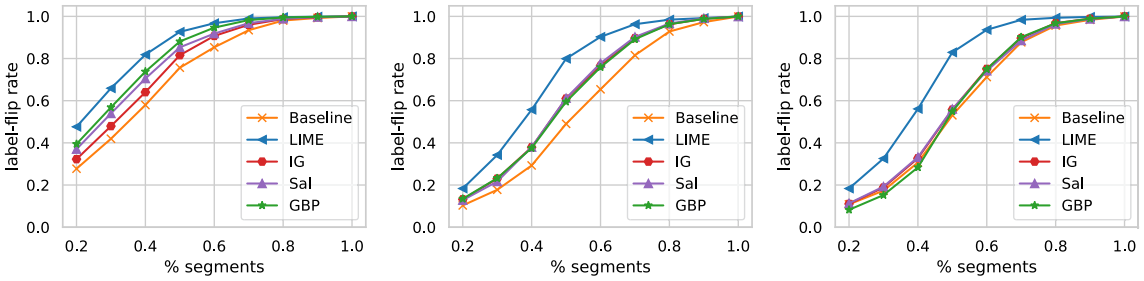
- *mean*: average over raw pixel-wise attributions per segment;
- *mean_abs*: average over the *absolute* pixel-wise attributions per segment;
- *mean_pos*: average over the *positive* pixel-wise attributions per segment.

An example for each of the aggregation approaches, along with an example of how Göpfert et al. [14] post-process attribution maps, is shown in Fig. 4.

In preliminary experiments, *mean_abs* led to the highest label-flip rates (cf. Sect. 5.1), which is why we report results using this approach below.

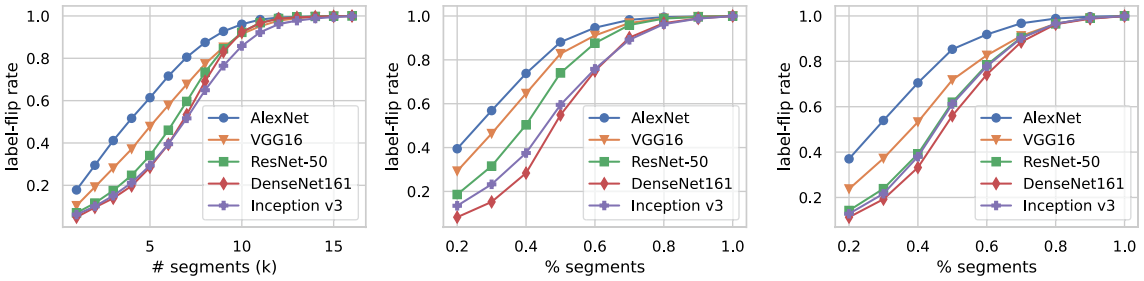
[4] Here: squares; we continue to refer to rectangles for consistency.

[5] This leads to *approximately* 16 segments per image, see Table 6 in Appendix.



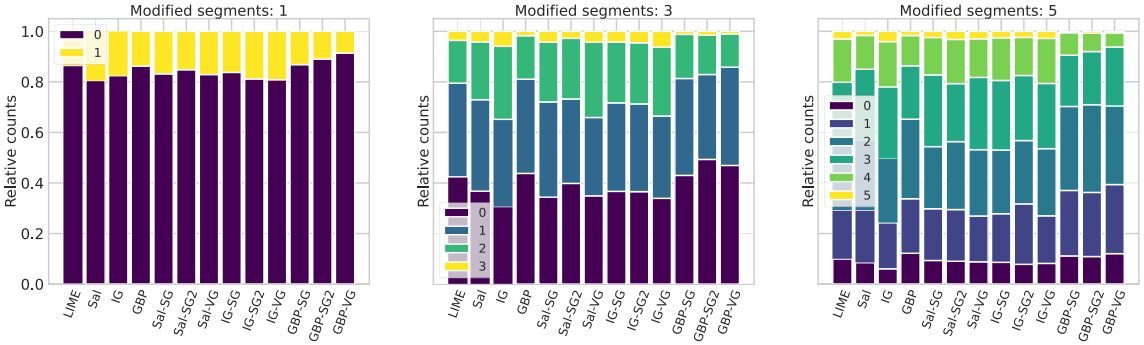
(a) AlexNet **(b)** Inception v3 **(c)** DenseNet161

Fig. 6 Label-flip rates of standard explainers vs. baseline on different models (SLIC segments) for different percentage of perturbed segments added

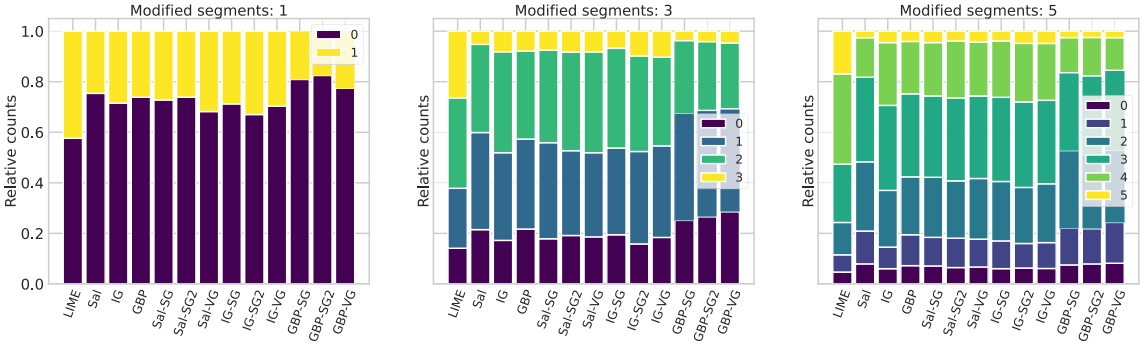


(a) LIME (rectangles) **(b)** GBP (SLIC) **(c)** Sal (SLIC)

Fig. 7 Label-flip rates of standard explainers for different models



(a) Architecture: Inception v3, segments: rectangles



(b) Architecture: Inception v3, segments: SLIC

Fig. 8 Percentage of correctly recovered k segments after adding $k \in \{1, 3, 5\}$ segments of a perturbation, for different explainers

6.2 Using a fixed number of interpretable features

Similar to the experiments in Sect. 5.1, we compute explanations for all adversarial examples, this time for 13 different explainers and two types of segmentation: rectangles and SLIC. For each explainer and segmentation, we fix $k = 3$ and add segments of perturbations coinciding with the most influential segments to the original input and count how often the label is changed to compute the label-flip rate. Results for explanations using SLIC segmentation are found in Tables 2 and 3. The label-flip rates for the rectangular segments are summarised in Appendix in Tables 7 and 8. The results show the following: (i) Explanations in the image domain appear more meaningful when using SLIC segments compared to rectangular segments for $k = 3$. (ii) When using rectangular segments, all standard explainers perform similarly across models, with GBP slightly outperforming the others for AlexNet, VGG16, and ResNet-50; and LIME slightly outperforming the others for DenseNet and Inception v3. (iii) Explainers generally perform best for AlexNet, followed by VGG16, and worst for DenseNet161. (iv) When using SLIC segments, LIME outperforms all other explainers by a large margin. The ranking of LIME, GBP, and Sal is consistent with the results in [14] (who analysed Inception v3) for AlexNet, VGG16, ResNet-50, and Inception v3. (v) Smoothing does not always help and in many cases it even decreases the performance. There is a noticeable pattern, however, when smoothing helps. For ResNet-50, DenseNet161, and Inception v3, smoothing increases—across all ensemble methods—the label-flip rate (i.e. the relative number of times parts of a perturbation change the original prediction of an input) for the setting with SLIC segments. (vi) In most cases, 3 segments are not sufficient to "explain" a prediction.

6.3 Varying the number of interpretable features

We also look at the label-flip rate of each explainer when increasing k and analyse the findings from the previous section in more detail. For rectangular segments, we perform this analysis for all $k \in \{1, ..., 16\}$. Due to the varying number of SLIC segments (cf. Table 6), we follow a different approach: we iteratively add $[20, 30, ..., 100]\%$ of all segments. Based on the findings in the previous experiment, we particularly want to investigate the following:

Rectangles vs. SLIC: For this experiment, we also add rectangular segments in percentage steps. Figure 5 shows the results for LIME, IG and Sal on ResNet-50. LIME works better when SLIC segments are used, while the other explainers work better with rectangles. This finding holds across all models.

Standard explainers vs. baseline: Figure 6 shows that in contrast with the audio domain, all explainers are at least on par with our proposed magnitude baseline. For AlexNet and Inception v3, all explainers slightly outperform the baseline, for DenseNet only LIME manages to do so.

Comparing different architectures: In Fig. 7, the performance of standard explainers for different model architectures is compared. As already suggested by the results in the previous section, it seems "easier" to explain predictions made by AlexNet as indicated by the higher label-flip rate across all explainers. VGG16 follows AlexNet in all comparisons. For the rest of the models, it is not as clear, but in general explainers perform (slightly) better for ResNet-50 and worst for Inception v3 and DenseNet161.

6.4 Explaining 'localised' perturbations

We also want to look at refined adversarial image perturbations. After splitting the perturbations into image segments corresponding to the segments used by our explainers, the k segments with the highest magnitude are chosen and used to perturb the original data. In Fig. 8, we show results on adversarial subsets of the data for which adding $k \in \{1, 3, 5\}$ adversarial segments change the original prediction of a network to any new classification[6].

As the results for the explainers vary only slightly for different settings, Fig. 8 shows exemplary results for Inception v3 on rectangular and SLIC segments (experiment performed on 10% of validation data). The x-axes show different explainers; the y-axes show, in different colours, the fraction of times a particular amount of segments (out of k) has been recovered by the explainer. For more experimental results, we refer to Appendix (Sect. E.1).

Figure 8 shows that all explainers we look at struggle to recover perturbed segments. In the extreme case, i.e. when perturbing only a single segment (left-most plots), the majority of segments is not detected (darkest colour), only improving slightly if we look at SLIC segments. For increasing k, the explainers tend to detect 1-3 (out of 3 or 5) perturbed segments (medium colours). In the rarest cases, all segments are recovered correctly. The main observable trends are that SLIC segments tend to be easier to detect

[6] Due to complexity, we report results on either 100% or 10% of the adversaries; preliminary experiments showed, however, similar trends.

except in the case of AlexNet (results omitted here) and that standard explainers, i.e. LIME, IG, Sal, and GBP, usually slightly outperform their more advanced counterparts.

7 Discussion and conclusion

In this work, we (i) investigated the performance of different explanation methods on detecting non-obvious causes for a prediction by using adversarial perturbations as a ground truth and (ii) compared these explainers to a new non-random baseline. We tested explainers in two different domains, namely for audio (singing voice detector) and images (different ImageNet classifiers). To treat perturbation- and gradient-based explainers fairly, we proposed a pixel-wise attribution aggregation, where different granularities are overcome by mapping all attributions to interpretable segments.

Our overarching result is that explainers struggle—regardless of the particular domain, task or model architecture—to find the parts of inputs that are really relevant for a system's prediction. We were able to demonstrate this empirically via a number of carefully designed experiments providing numerous counterexamples to one of the central claims of the field of explanation: being able to "explain[s] the predictions of any classifier in an interpretable and faithful manner" [17]. Our counterexamples disprove such universal statements concerning explanation methods. In more detail, our results show that:

- In the audio domain, the explanation method LIME is not able to recover perturbed segments in a satisfactory manner, with even the baseline performing better
- In the image domain, all investigated explanation methods at least surpass the baseline, but nevertheless are again not able to recover perturbed segments correctly
- In the image domain, explainers perform somewhat better in recognising perturbed segments when looking at "simpler" image classifier architectures
- For images, LIME performs best among the explainers (particularly when using SLIC segments), while standard gradient-based explainers (e.g. IG, Sal, GBP) usually outperform their more advanced versions

There are two major lessons from our work that could move the field of explainability forward: (i) In line with previous results [14], we recommend using ground-truth based on adversarial perturbations. This approach, and also more general dataset modifications (cf. [13]), will no longer rely on questionable ground-truth provided by human observers, avoid their confirmation bias (cf. [2]), and hence allow for a more objective comparison of methods. (ii) We recommend to always use and compare different explainers since our experiments indicate that methods perform differently depending on the application domain (cf. LIME performing below baseline for audio but clearly above for images). This is in line with previous results where explainers perform differently even for different datasets within the same domain [43].

Supplementary information Not applicable.

Appendix A Explanations

In this section, we show exemplary explanations for different ensemble explainers we compare in this work in Fig. 9. The figure shows an input image and how the nine different ensemble explainers would explain the (correct) prediction of the image.

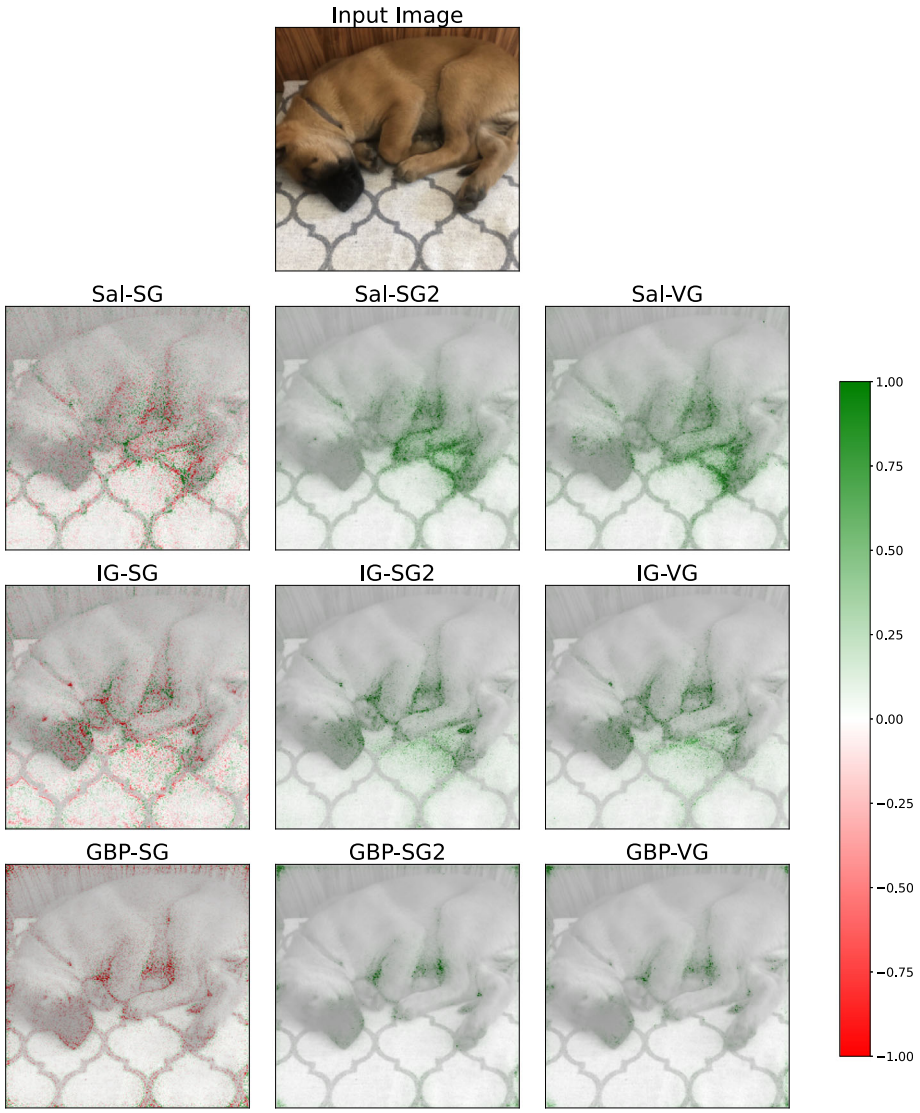


Fig. 9 Example explanations for each of the used ensemble explainers. Different colours depict the weights/attributions that different explainers assign to particular segments/pixels, with higher numbers indicating more, and lower numbers less "important" features for a prediction

Appendix B Adversarial attacks

B.1 Hyper-parameters

In this section, we list the exact hyper-parameters used to attack the image classifiers (Table 4) with PGD and the singing voice detector (Table 5) with the C&W-based attack.

Table 4 Hyper-parameters of PGD attack on different architectures. Settings were chosen based on most successful adversaries with least difference between clean and adversarial images within grid search

Architecture	ϵ	η
AlexNet	0.10	0.01
VGG16	0.05	0.01
ResNet-50	0.05	0.01
DenseNet161	0.05	0.01
Inceptionv3	0.05	0.01

Table 5 Hyper-parameters of C &W attack on SVDS. Setting was chosen based on the highest amount of successful attacks on the audio validation data

α	ϵ	η
15	0.1	0.0005

B.2 Exemplary adversaries

In this section, we show (five random) exemplary adversarial examples for each of the attacked model architectures in Figs. 10, 11 and 12.

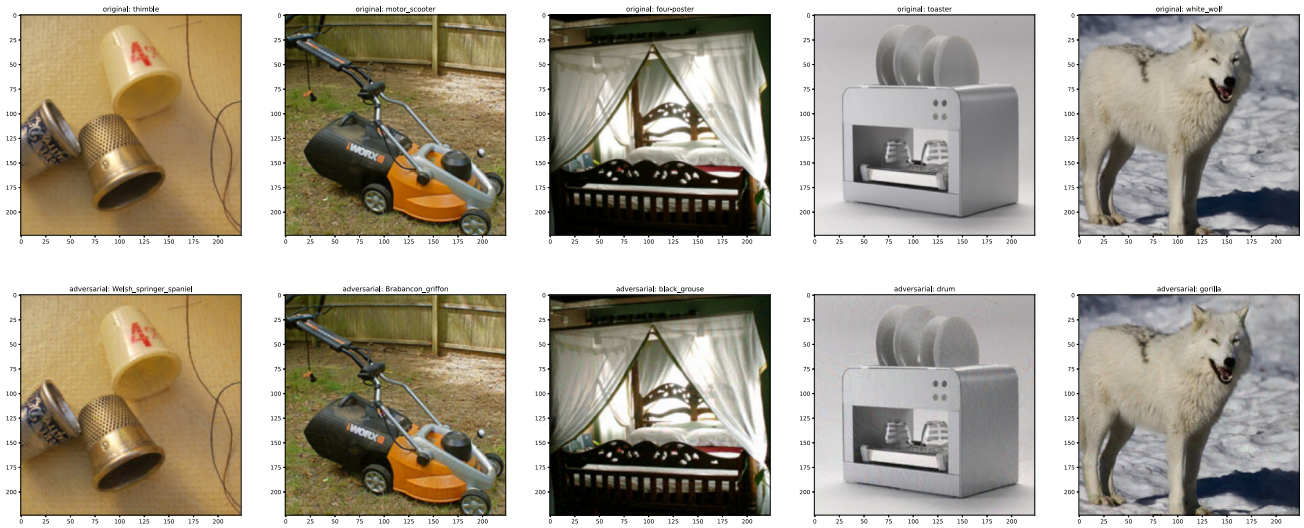
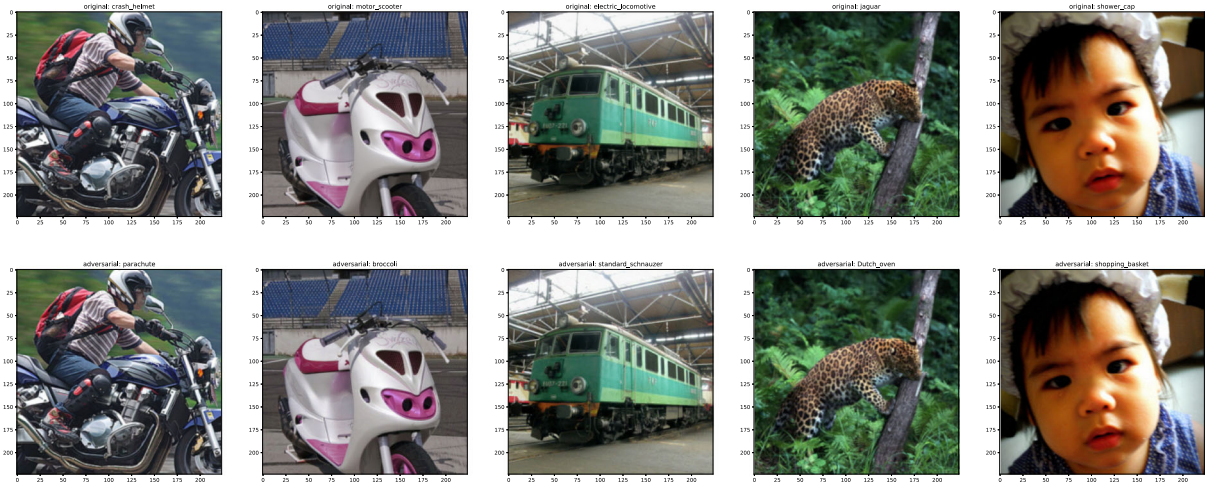
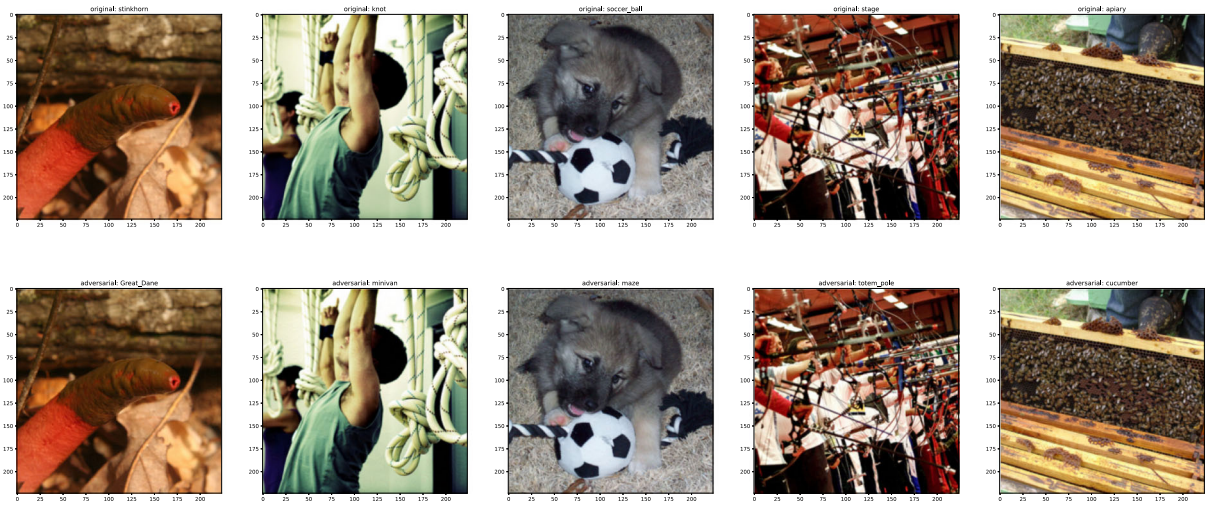


Fig. 10 5 Random examples of adversaries computed with a Projected Gradient Descent (PGD) attack for AlexNet. First row denotes original files, second row adversaries. Hyper-parameters for attacks are picked as explained in Sect. B.1

(a) Architecture: VGG16



(b) Architecture: ResNet-50



(c) Architecture: DenseNet161

◀**Fig. 11** 5 Random examples of adversaries computed with a PGD attack for VGG16, ResNet-50, and DenseNet161. First rows denote original files, second rows adversaries. Hyper-parameters for attacks are picked as explained in Sect. B.1

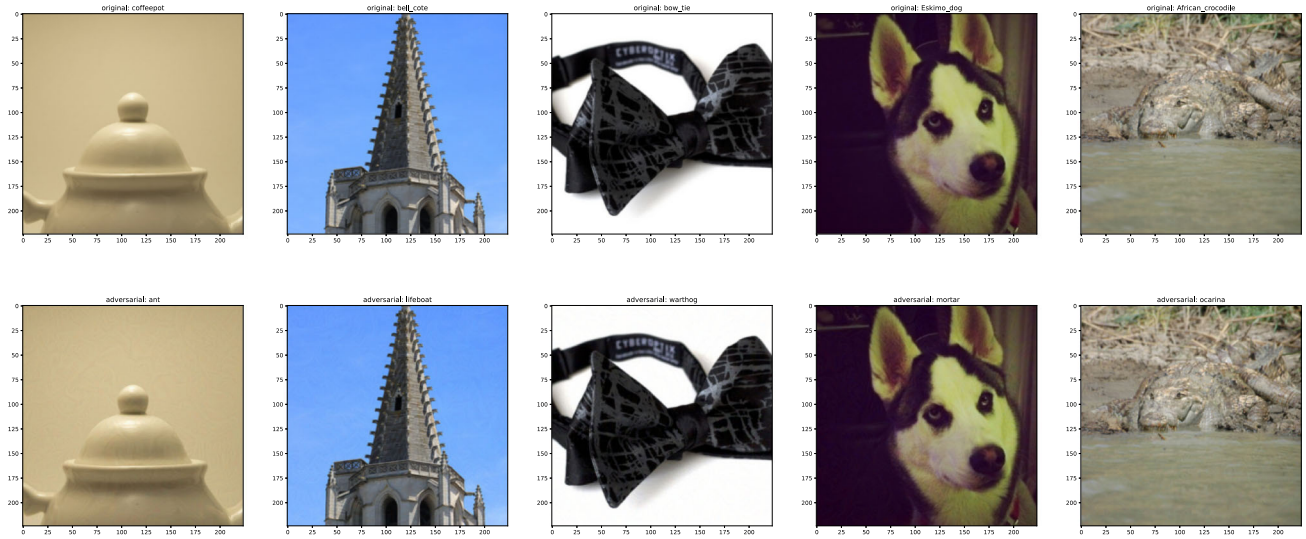


Fig. 12 5 Random examples of adversaries computed with a PGD attack for Inception v3. First rows denote original files, second rows adversaries. Hyper-parameters for attacks are picked as explained in Sect. B.1

Appendix C Implementation details

For all experiments on image data, we use pre-trained classifiers provided by `torchvision`[7]. In order to perform our experiments on a variety of different explainers, we use the library `captum`[8] to both implement the discussed explanation methods, as well as the PGD attack.

Appendix D SLIC segmentation

In this section, we provide a table with the average (± standard deviation) of the number of segments which are found by the SLIC algorithm. We compute these segments for all adversaries of different model architectures (Table 6).

Table 6 Average ± standard deviation of number of segments found by SLIC for adversaries of different models

Architecture	# Segments
AlexNet	11.039 ± 2.307
VGG16	11.048 ± 2.305
ResNet-50	11.042 ± 2.307
DenseNet161	11.046 ± 2.300
Inceptionv3	10.837 ± 2.384

Appendix E Additional results: image domain

In this section, we show additional results for the experiments in Sect. 6.2 and 6.4.

E.1 Using a fixed number of interpretable features

Here we show the omitted results from Sect. 6.2, in particular the label-flip rates for standard explainers (Table 7) and ensemble explainers (Table 8) for rectangular segments.

Table 7 Label-flip rates for standard explainers using rectangular segments and $k = 3$

Model	LIME	IG	Sal	GBP
AlexNet	41.14	38.90	41.52	43.80
VGG16	28.14	27.72	27.74	31.84
ResNet-50	17.52	15.56	15.36	17.60
DenseNet161	13.90	12.96	12.34	9.34
Inception v3	15.10	14.98	14.54	13.00

[7] https://pytorch.org/vision/stable/models.html.

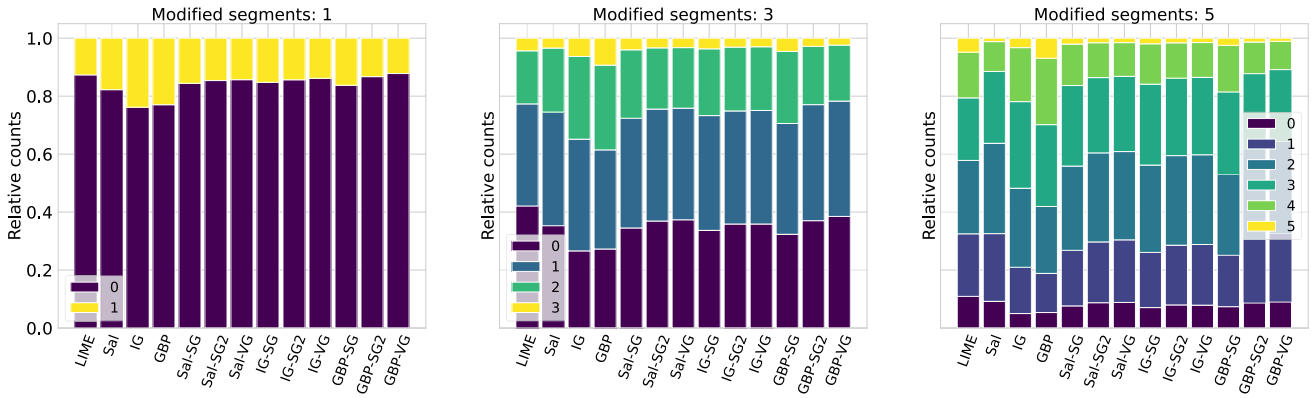
[8] https://captum.ai/.

Table 8 Label-flip rates for ensemble explainers using rectangular segments and $k = 3$

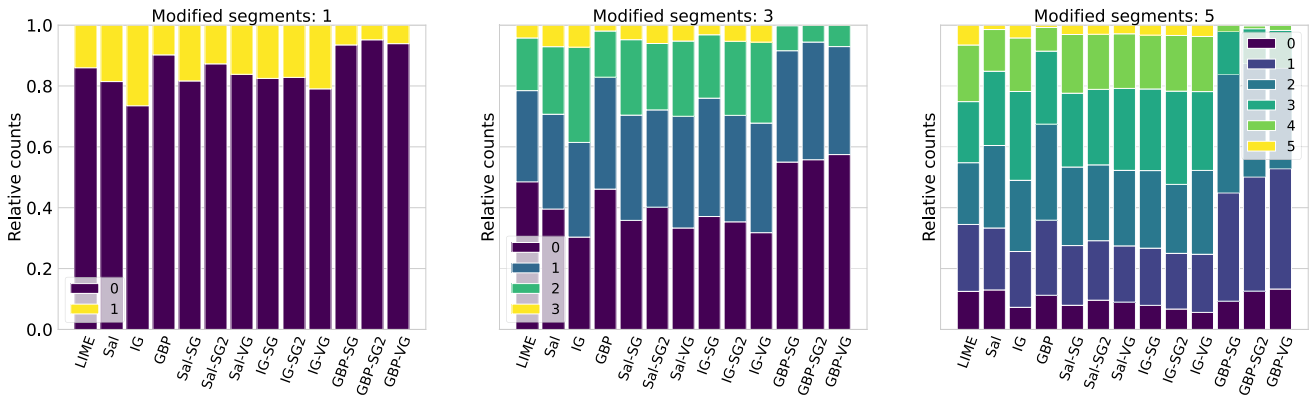
	Smoothgrad			Smoothgrad^2			Vargrad		
Model	IG	Sal	GBP	IG	Sal	GBP	IG	Sal	GBP
AlexNet	32.96	34.92	30.08	34.62	35.02	29.52	33.76	35.56	30.58
VGG16	24.36	24.30	21.60	23.74	25.28	22.08	25.24	24.86	21.82
ResNet-50	17.78	16.90	16.72	17.10	17.16	16.38	17.24	17.24	16.46
DenseNet161	13.92	14.44	6.48	13.66	14.90	6.20	14.22	14.22	6.62
Inception v3	17.80	17.64	13.36	17.62	18.20	11.24	17.52	17.94	11.12

E.2 Explaining "localised" perturbations

In what follows, results omitted in Sect. 6.4 are shown. We show an excerpt thereof, as the results vary only slightly. Figure 13 shows results when trying to recover a fixed number of perturbed segments with different explainers for two architectures for rectangular segments. Figure 14 shows the same for SLIC segments. All plots show the results when perturbing $k \in \{1, 3, 5\}$ segments; the x-axes show different explainers, the y-axes their success-rate in recovering a certain number out of all changed segments.

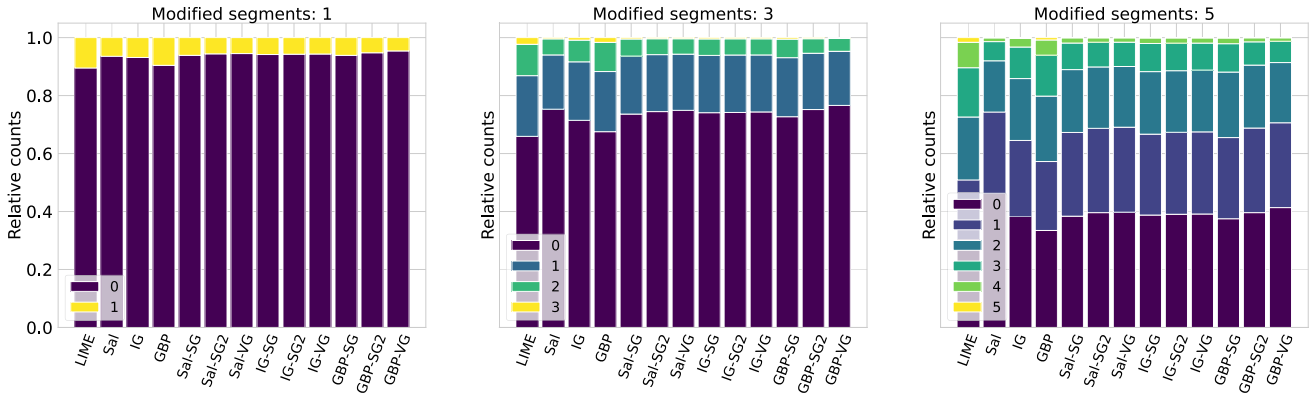


(a) Architecture: AlexNet (100% data)

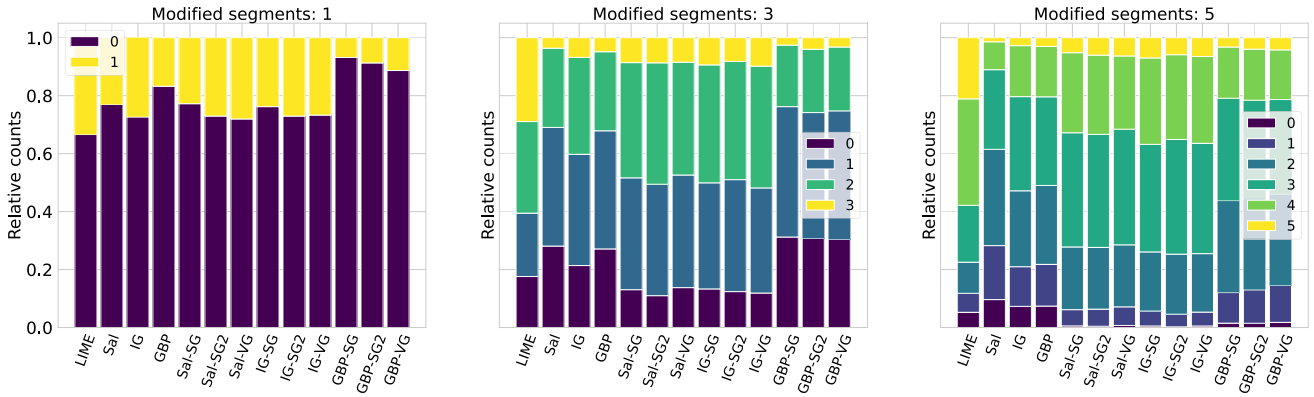


(b) Architecture: DenseNet161 (10% data)

Fig. 13 Percentage of correctly recovered k segments after adding $k \in \{1, 3, 5\}$ rectangular segments of a perturbation, for different explainers



(a) Architecture: AlexNet (100% data)



(b) Architecture: DenseNet161 (10% data)

Fig. 14 Percentage of correctly recovered k segments after adding $k \in \{1, 3, 5\}$ SLIC segments of a perturbation, for different explainers

Acknowledgements This research has received funding from the Austrian Science Fund (FWF, project number P 31988). GW's work is supported by the European Research Council (ERC) under the EU's Horizon 2020 research and innovation programme, grant agreement No 101019375 (*Whither Music?*). For the purpose of open access, the authors have applied a CC BY public copyright licence to any author accepted manuscript version arising from this submission.

Funding Open access funding provided by Austrian Science Fund (FWF).

Availability of data and materials The datasets analysed during the current study are available as Jamendo-VAD at https://www.ismir.net/resources/datasets (cf. [30]) and ImageNet at https://www.kaggle.com/competitions/imagenet-object-localization-challenge/data (cf. [32]).

Declarations

Conflict of interest The authors of the manuscript confirm that there are no conflict of interest to declare.

References

1. Lipton ZC (2018) The mythos of model interpretability. Commun ACM 61(10):36–43. https://doi.org/10.1145/3233231
2. Adebayo J, Gilmer J, Muelly M, Goodfellow IJ, Hardt M, Kim B (2018) Sanity checks for saliency maps. In: Advances in neural information processing systems 31: annual conference on neural information processing systems 2018, NeurIPS, pp 9525–9536
3. Neely M, Schouten SF, Bleeker, MJR, Lucic A (2021) Order in the court: explainable AI methods prone to disagreement. CoRR **abs/2105.03287**. https://doi.org/10.48550/arXiv.2105.03287
4. Krishna S, Han T, Gu A, Pombra J, Jabbari S, Wu S, Lakkaraju H (2022) The disagreement problem in explainable machine learning: a practitioner's perspective. CoRR **abs/2202.01602**. https://doi.org/10.48550/arXiv.2202.01602
5. Carvalho DV, Pereira EM, Cardoso JS (2019) Machine learning interpretability: a survey on methods and metrics. Electronics 8(8):832. https://doi.org/10.3390/electronics8080832

6. Praher V, Prinz K, Flexer A, Widmer G (2021) On the veracity of local, model-agnostic explanations in audio classification: targeted investigations with adversarial examples. In: Proceedings of the 22nd international society for music information retrieval conference, ISMIR, pp. 531–538. https://doi.org/10.5281/zenodo.5624471
7. Samek W, Binder A, Montavon G, Lapuschkin S, Müller K (2017) Evaluating the visualization of what a deep neural network has learned. IEEE Trans Neural Netw Learn Syst 28(11):2660–2673. https://doi.org/10.1109/TNNLS.2016.2599820
8. Hooker S, Erhan D, Kindermans P, Kim B (2019) A benchmark for interpretability methods in deep neural networks. In: Advances in neural information processing systems 32: annual conference on neural information processing systems 2019, NeurIPS, pp 9734–9745
9. Arras L, Osman A, Samek W (2022) CLEVR-XAI: a benchmark dataset for the ground truth evaluation of neural network explanations. Inf Fusion 81:14–40. https://doi.org/10.1016/j.inffus.2021.11.008
10. Yona G, Greenfeld D (2021) Revisiting sanity checks for saliency maps. CoRR abs/2110.14297. https://doi.org/10.48550/arXiv.2110.14297
11. Hase P, Bansal M (2020) Evaluating explainable AI: which algorithmic explanations help users predict model behavior? In: Proceedings of the 58th annual meeting of the association for computational linguistics, ACL, pp 5540–5552. Association for computational linguistics, Online. https://doi.org/10.18653/v1/2020.acl-main.491
12. Jesus SM, Belém C, Balayan V, Bento J, Saleiro P, Bizarro P, Gama J (2021) How can I choose an explainer?: an application-grounded evaluation of post-hoc explanations. In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, FAccT, pp 805–815. ACM, New York, NY, United States. https://doi.org/10.1145/3442188.3445941
13. Zhou Y, Booth S, Ribeiro MT, Shah J (2021) Do feature attribution methods correctly attribute features? CoRR abs/2104.14403. https://doi.org/10.48550/arXiv.2104.14403
14. Göpfert JP, Wersing H, Hammer B (2019) Recovering localized adversarial attacks. In: Artificial neural networks and machine learning - theoretical neural computation - proceedings of the 28th international conference on artificial neural networks, ICANN. Lecture notes in computer science, vol 11727, pp 302–311. Springer, Cham. https://doi.org/10.1007/978-3-030-30487-4_24
15. Simonyan K, Vedaldi A, Zisserman A (2014) Deep inside convolutional networks: visualising image classification models and saliency maps. In: Workshop track proceedings of the 2nd international conference on learning representations, ICLR
16. Springenberg JT, Dosovitskiy A, Brox T, Riedmiller MA (2015) Striving for Simplicity: the all convolutional net. In: Workshop track proceedings of the 3rd international conference on learning representations, ICLR
17. Ribeiro MT, Singh S, Guestrin C (2016) "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, pp 1135–1144. ACM, New York, NY, United States. https://doi.org/10.1145/2939672.2939778
18. Schlüter J, Lehner B (2018) Zero-mean convolutions for level-invariant singing voice detection. In: Proceedings of the 19th international society for music information retrieval conference, ISMIR, pp 321–326. https://doi.org/10.5281/zenodo.1492413
19. Mishra S, Sturm BL, Dixon S (2017) Local interpretable model-agnostic explanations for music content analysis. In: Proceedings of the 18th international society for music information retrieval conference, ISMIR, pp 537–543. https://doi.org/10.5281/zenodo.1417387
20. Chettri B, Sturm BL, Benetos E (2018) Analysing replay spoofing countermeasure performance under varied conditions. In: Proceedings of the 28th IEEE international workshop on machine learning for signal processing, MLSP, pp 1–6. https://doi.org/10.1109/MLSP.2018.8516968
21. Haunschmid V, Chowdhury S, Widmer G (2019) Two-level explanations in music emotion recognition. Machine learning for music discovery workshop, ML4MD at ICML2019. https://doi.org/10.48550/arXiv.1905.11760
22. Mishra S, Benetos E, Sturm BL, Dixon S (2020) Reliable local explanations for machine listening. In: Proceedings of the 2020 international joint conference on neural networks, IJCNN, pp 1–8. https://doi.org/10.1109/IJCNN48605.2020.9207444
23. Smilkov D, Thorat N, Kim B, Viégas FB, Wattenberg M (2017) SmoothGrad: removing noise by adding noise. CoRR abs/1706.03825. https://doi.org/10.48550/arXiv.1706.03825
24. Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Süsstrunk S (2012) SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans Pattern Anal Mach Intell 34(11):2274–2282. https://doi.org/10.1109/TPAMI.2012.120
25. Molnar C (2022) Interpretable machine learning, 2nd edn. christophm.github.io/interpretable-ml-book/
26. Sundararajan M, Taly A, Yan Q (2017) Axiomatic Attribution for Deep Networks. In: Proceedings of the 34th international conference on machine learning, ICML. proceedings of machine learning research, vol 70, PMLR, Sydney, NSW, Australia, pp 3319–3328
27. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2018) Towards deep learning models resistant to adversarial attacks. In: Proceedings of the 6th international conference on learning representations, ICLR
28. Prinz K, Flexer A, Widmer G (2021) On end-to-end white-box adversarial attacks in music information retrieval. Trans Int Soc Music Inf Retr 4(1):93–104. https://doi.org/10.5334/tismir.85
29. Carlini N, Wagner DA (2018) Audio adversarial examples: targeted attacks on speech-to-text. In: Proceedings of the IEEE security and privacy workshops, SP workshops, pp 1–7. IEEE computer society, Washington, DC, United States. https://doi.org/10.1109/SPW.2018.00009
30. Ramona M, Richard G, David B (2008) Vocal detection in music with support vector machines. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, ICASSP, pp 1885–1888
31. Schlüter J, Grill T (2015) Exploring data augmentation for improved singing voice detection with neural networks. In: Proceedings of the 16th international society for music information retrieval conference, ISMIR, pp 121–126. https://doi.org/10.5281/zenodo.1417745
32. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein MS, Berg AC, Fei-Fei L (2015) ImageNet large scale visual recognition challenge. Int J Comput Vision 115(3):211–252. https://doi.org/10.1007/s11263-015-0816-y
33. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the 2016 IEEE conference on computer vision and pattern recognition, CVPR, pp 2818–2826. IEEE computer society, Washington, DC, United States. https://doi.org/10.1109/CVPR.2016.308
34. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the 2016 IEEE conference on computer vision and pattern recognition, CVPR, pp 770–778. IEEE computer society, Washington, DC, United States. https://doi.org/10.1109/CVPR.2016.90

35. Krizhevsky A (2014) One weird trick for parallelizing convolutional neural networks. CoRR abs/1404.5997. https://doi.org/10.48550/arXiv.1404.5997
36. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: Bengio Y, LeCun, Y. (eds) Proceedings of the 3rd international conference on learning representations, ICLR
37. Huang G, Liu Z, van der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the 2017 IEEE conference on computer vision and pattern recognition, CVPR, pp 2261–2269. IEEE computer society, Washington, DC, United States. https://doi.org/10.1109/CVPR.2017.243
38. Zintgraf LM, Cohen TS, Welling M (2016) A new method to visualize deep neural networks. CoRR abs/1603.02518. https://doi.org/10.48550/arXiv.1603.02518
39. Zhou B, Khosla A, Lapedriza, À., Oliva A, Torralba A (2016) Learning deep features for discriminative localization. In: Proceedings of the 2016 IEEE conference on computer vision and pattern recognition, CVPR, pp. 2921–2929. IEEE computer society, Washington, DC, United States. https://doi.org/10.1109/CVPR.2016.319
40. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the 2017 IEEE international conference on computer vision, ICCV, pp 618–626. IEEE computer society, Washington, DC, United States. 10.1109/ICCV.2017.74
41. Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: Fleet DJ, Pajdla T, Schiele B, Tuytelaars, T. (eds.) Proceedings of the 13th European conference on computer vision, ECCV. Lecture notes in computer science, vol 8689, pp 818–833. Springer, Cham. https://doi.org/10.1007/978-3-319-10590-1_53
42. Ayhan MS, Kümmerle LB, Kühlewein L, Inhoffen W, Aliyeva G, Ziemssen F, Berens P (2022) Clinical validation of saliency maps for understanding deep neural networks in ophthalmology. Med Image Anal 77:102364. https://doi.org/10.1016/j.media.2022.102364
43. Gevaert A, Rousseau A, Becker T, Valkenborg D, Bie TD, Saeys Y (2022) Evaluating feature attribution methods in the image domain. CoRR abs/2202.12270. https://doi.org/10.48550/arXiv.2202.12270