

# Almost for Free: Crafting Adversarial Examples with Convolutional Image Filters

Alexander Warnecke<sup>1</sup> Konrad Rieck<sup>1</sup>

## Abstract

Adversarial examples in machine learning are typically generated using gradients, obtained either directly through access to the model or approximated via queries to it. In this paper, we propose a much simpler approach to craft adversarial examples, drawing inspiration from insights of explainable machine learning. In particular, we design *adversarial image filters* that are based on classic edge detection algorithms but optimized to deceive learning models. The resulting untargeted attacks are transferable and require only a single pass over the input. Empirically, we find that  $3 \times 3$  filters already enable success rates between 30% and 80% on different neural networks. Compared to related approaches using generative models for crafting adversarial examples, we reduce the number of parameters by five orders of magnitude, resulting in a very efficient attack. When investigating the parameters of the learned filters, we observe interesting properties such as a high transferability between models and structures common to classic image filters. Our results provide further insights into the vulnerability of neural networks and their fragility to malicious noise.

## 1. Introduction

Deep neural networks achieve remarkable performance in various learning tasks, particularly in the vision domain (Krizhevsky et al., 2012; Simonyan & Zisserman, 2015). These successes, however, are overshadowed by their vulnerability to adversarial examples—deceptive inputs designed to mislead predictions (Szegedy et al., 2014). The prevalent paradigm for constructing these attacks rests on using gradients of the learning model, obtained either through direct access to its parameters (Carlini & Wagner, 2017; Goodfellow et al., 2015) or by approximation through multiple queries to it (Chen et al., 2017; Ilyas et al., 2018).

<sup>1</sup>BIFOLD & TU Berlin

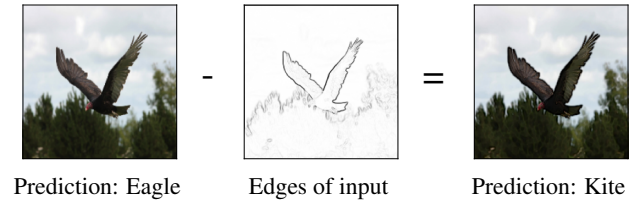


Figure 1. Example of adversarial image filters. An input image (left) is perturbed using a convolutional filter (middle) to generate an adversarial example (right).

In this paper, we deviate from this paradigm and propose *adversarial image filters* as a simple strategy for crafting adversarial examples. These filters deceive learning models through a single convolution and hence do not require the gradient information. Our approach is inspired by the connection between edges and gradients observed in explainable machine learning (Adebayo et al., 2018; Ancona et al., 2018). Starting from a classic Sobel filter for edge detection (Sobel & Feldman, 1973), our attack optimizes the convolution weights of the filter to maximize mispredictions of learning models. The resulting untargeted attack requires only a single pass over the input and transfers well between models. An example of our attack can be seen in Figure 1, where a single convolution induces a misclassification.

We empirically evaluate our attack with different convolutional filter sizes and find that a small  $3 \times 3$  filter already achieves attack success rates between 45% and 94% against learning models such as ResNet50 (He et al., 2016). The visual changes are hardly noticeable, remaining above a PSNR of 20. In comparison to related approaches that use generative models to create adversarial examples (Moosavi-Dezfooli et al., 2016a; Baluja & Fischer, 2018; Xiao et al., 2018), our attack reduces the number of parameters by five orders of magnitude. This substantial reduction not only enhances efficiency but drastically simplifies the overall attack mechanism. Furthermore, the learned filters demonstrate better performance when compared to attacks based on conventional image filters (Agarwal et al., 2020; Guo et al., 2020). While blurring filters can naturally induce mispredictions, they cannot achieve similar attack performance at the same PSNR level.

When investigating the parameters of the learned  $3 \times 3$  filters, we observe interesting properties: the filters are characterized by a high level of transferability across models. Due to their small size, model-specific properties can hardly be encoded in their weights, resulting in a largely model-independent attack. While the learned weight structure shares similarities with existing filters, such as the Laplacian of Gaussian filter, its specific configuration is unique and forms a new filter type. This filter type offers new view on the fragility of neural networks to malicious noise. By perturbing image regions with a single convolution, we achieve a notable decline in model performance, suggesting that complex approaches are not always necessary for effective attacks.

It is clear that a simple convolutional filter cannot compete with sophisticated optimization strategies for crafting adversarial examples. Still, our attack can serve as a lower bound on the performance of untargeted attacks. The run-time and memory complexity of more advanced strategies need to be weighed against our filters to put their performance improvements into perspective.

## 2. Related Work

The phenomenon of adversarial examples (AEs) in deep learning was first discovered by Szegedy et al. (2014) and has since become an active research area, resembling a cat-and-mouse game where new defenses are swiftly proposed and then quickly broken. In the case where model parameters are available to the adversary (white-box setup), AEs can be constructed using gradient information (e.g. Carlini & Wagner, 2017; Goodfellow et al., 2015; Papernot et al., 2016a; Moosavi-Dezfooli et al., 2016b). In the case where parameters are unavailable (black-box setup), various gradient approximations can be employed, such as shadow models (Papernot et al., 2016b), finite differences (Chen et al., 2017), Gaussian noise averages (Ilyas et al., 2018), evolution strategies (Ilyas et al., 2019), and random navigations through the input space (Guo et al., 2019). Even when only the network’s decision, rather than softmax scores, is available, strategies for crafting adversarial examples are available (Chen & Jordan, 2019; Brendel et al., 2018).

There are only a few approaches to crafting AEs that fall outside these two attack scenarios. Moosavi-Dezfooli et al. (2016a) introduce *universal adversarial perturbations*, which involve a *single* perturbation vector that can be added to *any* image to cause a misclassification. Similarly, Baluja & Fischer (2018) train a neural network to generate an adversarial perturbation for a given input using an adversarial criterion in the loss function. This approach shares similarities with adversarial filters, yet the trained networks require between 3.4 and 233.7 million parameters for a successful attack. The underlying approach has been further adapted

to generative models (Xiao et al., 2018; Poursaeed et al., 2017), most prominently generative adversarial networks (GANs) (Goodfellow et al., 2014), which transform a random noise vector into an adversarial perturbation. As part of this transformation, properties like transferability can be directly incorporated into the learning problem (Nakka & Salzman, 2021). However, the resulting architectures are complex, again involving a large number of model parameters.

Closest to our work are attacks that rely on image transformations, leveraging the sensitivity of neural networks to edges and structures. For example, Geirhos et al. (2018) maliciously replace textures in images. Similarly, Agarwal et al. (2020) and Guo et al. (2020) discuss the use of image filters for blur and motion blur to craft adversarial examples. Shamsabadi et al. (2020) propose a special type of adversarial example where contours are enhanced by decomposing the image into structural and residual parts using a neural network. He et al. (2023) show that the robustness of neural networks increases when using these examples for adversarial training. While our approach is related to this work, the concept of training an adversarial filter has not been explored so far. As we discuss in the evaluation, better performance can be attained if the convolution is specifically adapted to this malicious task.

## 3. Motivation and Background

We begin by defining the notation used throughout the paper and motivating our approach to generating adversarial examples with convolutional filters.

Let  $\theta \in \mathbb{R}^m$  represent a learning model with  $m$  parameters, and let  $f_\theta$  denote the associated prediction function. For an input  $\mathbf{x} \in \mathbb{R}^d$ , the model yields a prediction  $f_\theta(\mathbf{x}) \in \{1, \dots, c\}$ , where  $c$  denotes the number of classes. Given a labeled dataset  $D$  consisting of training examples, the optimal model parameters  $\theta^*$  are determined by minimizing the empirical risk as follows

$$\theta^* = \operatorname{argmin}_{\theta \in \mathbb{R}^m} \sum_{(\mathbf{x}, y) \in D} \ell(\mathbf{x}, y, \theta), \quad (1)$$

where  $\ell(\mathbf{x}, y, \theta)$  is a loss function that measures the difference between a prediction  $f_\theta(\mathbf{x})$  and the true label  $y$ . In the remainder of this paper we use public models that represent solutions of Equation (1) for benchmark datasets.

An untargeted adversarial example is an input  $\mathbf{x}' = \mathbf{x} + \delta$  that results from adding an imperceptible perturbation  $\delta \in \mathbb{R}^d$  to  $\mathbf{x}$  and has the property that  $f_\theta(\mathbf{x}) \neq f_\theta(\mathbf{x}')$  (Szegedy et al., 2014). Generating minimal perturbations invisible for the human eye can be defined as solutions of the optimization problem

$$\min_{\delta \in \mathbb{R}^d} \|\delta\|_p \quad \text{s.t.} \quad f_\theta(\mathbf{x} + \delta) \neq f_\theta(\mathbf{x}), \quad (2)$$

for some  $L^p$  norm,  $p \in \{1, 2, \infty\}$ . If we denote by  $\mathcal{A}$  an algorithm that generates adversarial examples, existing approaches can be broadly categorized using the function signatures of  $\mathcal{A}$ :

**White-box attacks** If the model parameters are available,  $\mathcal{A}$  is given by  $\delta = \mathcal{A}(\mathbf{x}, \theta)$ , that is, we know the model architecture and its parameters  $\theta$ . An example for such an approach is the FGSM algorithm (Goodfellow et al., 2015) that leverages first-order optimization techniques and solves Equation (2) by performing update steps of the form

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \tau \cdot \text{sgn}(\nabla_{\mathbf{x}} \ell(\mathbf{x}^{(t)}, f_{\theta}(\mathbf{x}^{(t)}), \theta)), \quad (3)$$

for a loss function  $\ell$ , a small learning rate  $\tau > 0$  and a data point  $\mathbf{x} = \mathbf{x}^{(0)}$ . The advantage of white-box algorithms is the high quality of the outcome and the fact that an AE can almost always be found. On the other hand, access to the model architecture and parameters is a strong assumption for real-world attacks.

**Black-box attacks** Generating adversarial examples in a black-box setting corresponds to the function signature  $\delta = \mathcal{A}(\mathbf{x}, f_{\theta})$ , where  $f_{\theta}$  is treated as an *oracle* function where we neither know the architecture, nor the parameters of the underlying model. Solutions to this approach usually try to approximate white-box algorithms by estimating the gradient  $\nabla_{\mathbf{x}} f$ , which constitutes the direction into which we have to move the input  $\mathbf{x}$ . A straight-forward approach is given by a finite difference estimation given by

$$\frac{\partial f_{\theta}(\mathbf{x})}{\partial x_j} \approx \frac{f_{\theta}(\mathbf{x} + \beta \mathbf{e}_j) - f_{\theta}(\mathbf{x} - \beta \mathbf{e}_j)}{2\beta} \quad (4)$$

for example where  $\beta > 0$  is a small step size and  $\mathbf{e}_j$  denotes the  $j$ -th unit vector in  $\mathbb{R}^d$  (Chen et al., 2017). Although black-box algorithms can yield excellent results, they require a lot of queries to the oracle which may be rejected by the application in practice (Debenedetti et al., 2024).

**Transformation attacks.** Interestingly, the process of crafting adversarial examples can also be described as a transformation that receives an input and returns its perturbed version. The signature is given by  $\delta = \mathcal{A}(\mathbf{x}, T)$ , where  $T : \mathbb{R}^d \rightarrow \mathbb{R}^d$  is a function that maps an input  $\mathbf{x}$  to its adversarial version  $\mathbf{x}'$ . In the literature,  $T$  has been chosen as either another neural network directly transforming the input (Baluja & Fischer, 2018) or as a generative adversarial network (GAN) that transforms a random image into a perturbation  $\delta$  given  $\mathbf{x}$  (Xiao et al., 2018). These approaches are considered gray-box attacks, as  $T$  is a learning function with a set of parameters  $\theta_T$  that must be optimized. Consequently, a dataset  $D_T$  from the same distribution as  $D$  is required, along with access to the model parameters to incorporate the adversarial criterion into the learning problem.

Once optimized, however, these approaches can generate adversarial examples (AEs) for any input, which is a useful property for concepts like adversarial training (Madry et al., 2018).

In this paper, we propose two transformation attacks related to edge detection from computer vision that employ convolutional filters as basis for the underlying transformation function.

**Model-independent attacks.** For our first new attack, we extend the gray-box assumptions to a case where  $T$  is completely independent of the model, that is, it neither uses model parameters nor oracle predictions. Our motivation for constructing adversarial examples in this manner stems from the work of Adebayo et al. (2018). They demonstrate that approaches to explain neural networks through saliency maps (Bach et al., 2015; Sundararajan et al., 2017) share similarities with simple edge detection. For intuition, Figure 2 shows explanations from the LRP explanation method (Bach et al., 2015) and edges generated by a simple Sobel filter (Sobel & Feldman, 1973) for inputs to a VGG13 network pre-trained on the ImageNet dataset. It is striking that the information provided by the explanation, although somewhat more nuanced, significantly overlaps with the edges extracted from the image.

Concurrently, Ancona et al. (2018) point out that—under certain conditions—explanations are equivalent to the gradient  $\nabla_{\mathbf{x}} f_{\theta}(\mathbf{x})$ , the core building block of white-box algorithms for crafting adversarial examples. Consequently, we ask: *If edges have similarities with the gradients  $\nabla_{\mathbf{x}} f_{\theta}(\mathbf{x})$ , could edge detection filters be adapted to construct adversarial examples?*

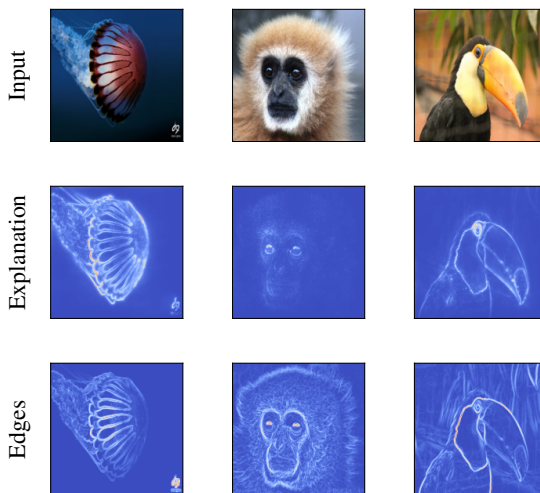


Figure 2. Input images (top row) with LRP explanations (middle row) and edges (bottom row) detected by a Sobel filter.

Combining the observations from above we can define a strategy to craft model independent examples as

$$\delta = \mathcal{A}(\mathbf{x}) = \rho(\mathcal{E}(\mathbf{x})), \quad (5)$$

where  $\mathcal{E}$  is an edge detection method and  $\rho$  is a processing function. As a simple example, consider the function

$$\rho(\mathcal{E}(\mathbf{x})) = \pm\mu\mathcal{E}(\mathbf{x}), \quad (6)$$

where  $\mu \in \mathbb{R}$  is a scalar. Here, we aim to confuse the learning model by perturbing the edges in the image. If  $\mathcal{E}(\mathbf{x}) \approx \nabla_{\mathbf{x}}f_{\theta}(\mathbf{x})$  and  $\mu > 0$ , this rule can be seen as a single approximation step in the FGSM algorithm (Goodfellow et al., 2015). If  $\mu < 0$ , this procedure simply removes the information contained in the edge pixels by shifting their value closer to zero. In both cases, it is important to assure that the output of  $\rho$  remains a valid image, for example, by applying a suited clipping function to the outcome.

**Model-dependent attack** As a second approach, we return to the original setting of transformation attacks but use convolutional filters as the learning model. Specifically, we optimize their values based on a small training dataset with access to  $f_{\theta}$ . As we will see in Section 4.2, this approach yields better results compared to the first strategy but produces slightly less imperceptible and less successful perturbations compared to other gray-box strategies. However, due to the small size of the filters, the computational complexity is significantly reduced, which raises questions about the necessity of complex learning models for transformation attacks.

## 4. Approach

In the following section, we formulate two different versions of our filter-based attack. One that is based on conventional edge filters and one that optimizes a given filter using some data similar to the training set of the target model, resulting in an adversarial filter.

Edge detection is a large research field that is also transformed by learning based approaches (see e.g. Sun et al., 2022, for an overview) but we will investigate implementations of Equation (6) using classic approaches from computer vision that are based on spatial kernels. Some examples for such filters are given in Figure 3 and illustrate that basic tasks like image smoothing (Filters (a) and (b)), image sharpening (Filter (c)) and combinations thereof (Filter (d)) can be computed with a single convolution operation. These approaches are light weight, easy to implement and allow for a direct investigation of the outcome.

### 4.1. Edge Filter Attack

We start with an implementation of Equation (6) based on a Sobel filter, a classic approach from computer vision to detect edges in images (Sobel & Feldman, 1973). Given a gray-scale image  $\mathbf{x} \in \mathbb{R}^{w \times h}$ , the Sobel filter computes a convolution between  $\mathbf{x}$  and its spatial filters  $K_x$  and  $K_y$  given by

$$K_x = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad K_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}. \quad (7)$$

Afterwards, the the gradient magnitude at each pixel is computed by

$$S(\mathbf{x}) = \sqrt{(\mathbf{x} * K_x)^2 + (\mathbf{x} * K_y)^2}. \quad (8)$$

The outcome is thus a gray-scale image itself and the perturbation added to each color channel of the input image is identical. Intuitively, the Sobel filter detects changes in the horizontal direction with  $K_x$ , in the vertical direction by  $K_y$  and combines their magnitude. This simple filter is related to the Prewitt filter (Prewitt, 1970) and the Scharr filter (Scharr, 2014), which change the parameter values. Extensions to a filter sizes of  $5 \times 5$  and more accurate approximations (Kekre & Gharge, 2010) also exist, however, for the sake of simplicity, we stick with the Sobel filter to evaluate our edge filter attack for generating adversarial examples.

### 4.2. Adversarial Filter Attack

The previous attack is based on the idea that the edges are approximately equal to the gradient of the classification result with respect to the input. Since the employed edge filters are based on a convolution operation, we can ask whether there exist better filters to craft adversarial examples. In other words, is there a filter that particularly impairs the performance of learning models in computer vision?

Given a convolutional filter  $\mathbf{k}$  we can search for its optimal values by formulating an optimization problem given by

$$\operatorname{argmin}_{\mathbf{k}} - \sum_{\mathbf{x} \in D} \ell(\mathbf{x} + \mathbf{x} * \mathbf{k}, f_{\theta}(\mathbf{x}), \theta) + \lambda \|\mathbf{k}\|_2^2. \quad (9)$$

In order to solve this problem we require a dataset  $D$  that is representative for the training data and by definition this problem aims to craft untargeted adversarial examples since we only increase the loss on the original label  $f_{\theta}(x)$ . The  $L^2$  norm of the filter regularizes the perturbation strength, the loss term steers the success rate of the attack and the parameter  $\lambda$  balances both properties. If the learning model and the loss function are differentiable with respect to  $\mathbf{x}$

Smoothing	Gaussian	Laplacian	Laplacian of Gaussian	Adversarial Filter (ours)
$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$	$\begin{bmatrix} -0.54 & 0.80 & -0.53 \\ 0.80 & -1.00 & 0.80 \\ -0.51 & 0.81 & -0.53 \end{bmatrix}$
(a)	(b)	(c)	(d)	(e)

Figure 3. Examples for spatial filters that can be convolved with an image to transform it in various ways.

we can initialize the filters with random values and apply optimization schemes like SGD to solve the problem above. An example for a resulting filter is shown in Figure 3 (e) and we will discuss its properties and relations to other filters in Section 5.2.2.

Notice that we refrain from using the magnitude of the convolution outcome as in Equation (6) in order to allow negative values in  $\delta$  that may help to achieve the goal easier. It might look like a natural extension to compute the perturbation with multiple filters  $\mathbf{k}_1, \dots, \mathbf{k}_M$ , i.e.,  $\delta = \sum_{i=1}^M \mathbf{x} * \mathbf{k}_i$ , however, due to the linearity of the convolution operator, the same perturbation can be achieved with a single filter summing up the values of all filter. It is also straight forward to minimize the loss towards a target class  $y_t(\mathbf{x})$  that can be chosen by the attacker such that targeted adversarial examples can be generated by optimizing

$$\operatorname{argmin}_{\mathbf{k}} \sum_{\mathbf{x} \in D} \ell(\mathbf{x} + \mathbf{x} * \mathbf{k}, y_t(\mathbf{x}), \theta) + \lambda \|\mathbf{k}\|_2^2.$$

In consequence, we obtain two new attack strategies that significantly reduce the knowledge and capabilities of the adversary. The first strategy is readily applicable and does not require any knowledge of the model and the input sample. The second strategy requires white-box access to the model. Due to the few parameters of the filter, however, a considerably lower number of training examples is needed to achieve good results.

## 5. Evaluation

We evaluate both attack strategies on three neural networks, namely VGG-13 (Simonyan & Zisserman, 2015), ResNet-50 (He et al., 2016) and Inception-V3 (Szegedy et al., 2016) that were trained on the ImageNet dataset (Deng et al., 2009). In order to simulate an attack during test time, we use the ImageNet V2 dataset (Recht et al., 2019) that is composed of 10 000 images disjoint of the ImageNet dataset. We test our attack scheme on 5 000 randomly chosen images thereof and use the remaining 5 000 to optimize the filter in the adaptive attack. To evaluate the attack performance we use the attack success rate, i.e., the fraction of images for which  $f_\theta(\mathbf{x}) \neq f_\theta(\mathbf{x} + \delta)$ .

As a quantification of the stealthiness of  $\delta$  we employ the *peak signal-to-noise ratio*, a measure of difference between  $\mathbf{x}$  and  $\mathbf{x} + \delta$ . In our case, the PSNR thus becomes a function of  $\delta$  and is defined by

$$\text{PSNR}(\delta) = 10 \cdot \log_{10} \left( 3wh \cdot \frac{I_{\max}^2}{\|\delta\|_2^2} \right),$$

where  $I_{\max}$  is the maximum value a pixel can have, i.e., 255 in our experiments. The PSNR is measured in dB and usually applied for evaluating image compression methods, thus a low PSNR indicates higher information loss or a stronger perturbation in our case.

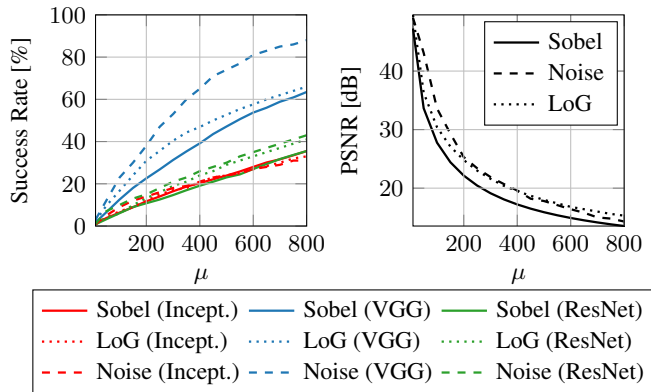


Figure 4. Success rate of the edge filter attack for Sobel- and Laplacian-of-Gaussian (LoG) filter (left) and PSNR between original image and the outcome of the attack (right). The dashed line indicate an attack where amplified Gaussian noise is added to the image such that the PSNR (left) or success rate (right) is equal to the corresponding Sobel filter attack outcome.

### 5.1. Edge Filter Attack

Figure 4 shows the evolution of the average attack success rate (left) and the PSNR (right) when increasing  $\mu$  in the attack scheme above. During our experiments we find that subtracting edge information ( $\mu < 0$ ) and adding it ( $\mu > 0$ ) achieves a similar performance with a slight advantage for the subtraction, therefore we present only the results for  $\mu < 0$ . As expected, the success rate of the attack rises at the cost of a lower PSNR as  $\mu$  is increased. The VGG model

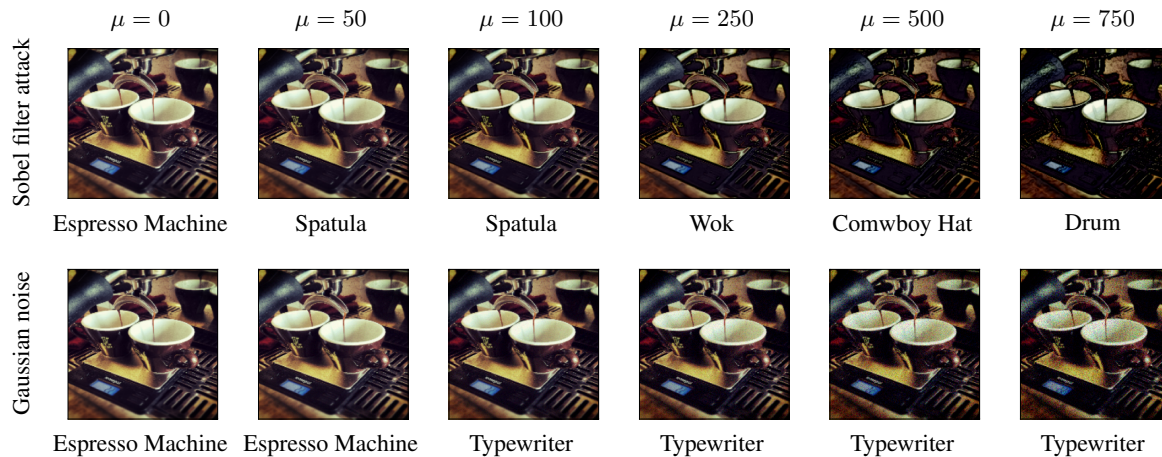


Figure 5. Outcome of the Sobel attack for different magnitudes  $\mu$ . The classification result is based on a VGG-13 model and is shown under the images. The Gaussian noise is computed such that the success rate is equal to the Sobel attack.

is clearly more prone to our attack, whereas the ResNet and Inception models behave very similar. To get a feeling for the strength of the perturbations, we present a trajectory of the attack for the VGG model when increasing  $\mu$  in Figure 5. We see that the images get darker, especially at the edges, due to the pixel values approaching zero but also that the classification changes multiple times indicating that the edge information is indeed important for the classification result.

As a baseline, we perform a simple attack where amplified Gaussian noise is subtracted from the input image, i.e.  $\delta = -t \cdot |\mathcal{N}(0, I)|$  for some  $t \in \mathbb{R}_+$ . Here we use the absolute value of the noise since the edge filter attack subtracts positive values due to the usage of the magnitude (see Equation (8)). Given a PSNR value of the Edge attack for some  $\mu$ , we can perform a binary search over  $t$  to find a value that achieves an equal PSNR value to compare the success rates at  $\mu$ . Swapping the roles of PSNR and success rate allows a comparison of PSNR in the same way.

The corresponding curves are presented with dashed lines in Figure 4 and the resulting images are also depicted in Figure 5. We see that the noise addition is better for the VGG model, equal for the ResNet model, and slightly worse for the Inception model regarding both, success rate and PSNR. This result should, however, be taken with a grain of salt since the noise is applied uniformly and changes every single pixel in the image indiscriminately, whereas the edge attack is restricted to small fractions thereof, see the large white area in Figure 1 for example. Still, crafting adversarial examples by subtraction of edge information is possible but achieving high success rates is difficult. At the same time, this simple attack required only 27 parameters in total and still was able to fool all three networks with adversarial examples that were computed in a "one shot", training-free manner, requiring no iterative optimization.

## 5.2. Adversarial Filter Attack

The adversarial attack has multiple parameters that influence the result, e.g. size of the filters or the number of training points in  $D$ . To allow a close comparison to the static attack investigated above and to be able to interpret the resulting filters, we use small filters of size three or five and optimize Equation (9) using SGD for 25 epochs. We further split the 5 000 images we have not been used so far into 3 000 images that can be used for training and 2 000 that will be used for validation. The 5 000 examples used for the Sobel attack serve as our test set such that both attacks are compared on the same data.

The regularization strength  $\lambda$  is the dominating parameter in the adversarial filter attack since it determines a success rate and a PSNR value that can be achieved during the optimization process. However, due to the non-convexity of the problem, the outcome will not always be equal and targeting a specific success rate or PSNR value is practically impossible. To this end, we optimize Equation (9) for multiple values for  $\lambda$ . Concretely, we sample values between  $\lambda = 0$ , i.e., high success rate and low PSNR, and  $\lambda = 0.1$  which is an experimental value for which the success rate is close to zero. Sampling enough values allows to pick a given PSNR value for all networks, for example, such that the success rates can be compared. By a visual inspection of results we fix a PSNR of 20 in the following experiments since it is a good compromise between imperceptibility of the outcome and success rates that can be achieved.

Figure 6 shows the success rates of the adversarial filter attack when varying the number of training examples between 100 and 3 000 and for convolution kernels of size  $3 \times 3$  and  $5 \times 5$ . We observe that this attack achieves much better results compared to the simple Edge filter attack: For the VGG model, we can increase the success rate from 27 % to

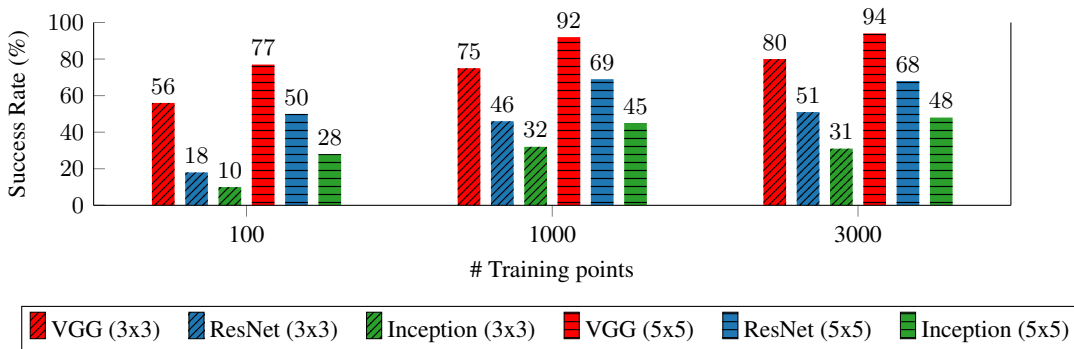


Figure 6. Success rate of the adaptive filter attack for different numbers of training examples and filter sizes. The PSNR is fixed to 20 for all models.

56 % with only 100 training examples. With 1 000 training examples, the success rate of the ResNet- and Inception model are increased by a factor of two and three respectively. Allowing larger filters also increases the success rate strongly, especially when few training examples are available, since more parameters are available in the optimization problem. Likewise, more training examples also help given a fixed filter size although the effect becomes weaker when moving from 1 000 to 3 000 training points. In summary, we can conclude that we can craft adversarial examples with adaptive filters in an extremely efficient way once a suitable dataset for optimizing the filters is available. Moreover, compared to similar approaches (e.g. Baluja & Fischer, 2018), the number of parameters is reduced by up to six orders of magnitude.

### 5.2.1. ADVERSARIAL CLASS DISTRIBUTION

Since our attack is performed in an untargeted way, it is worth investigating the classes, to which the adversarial versions of the images are assigned by the networks. Table 1 shows the five most common outcome classes of successful attacks for the ResNet model for two PSNR values where the filters are optimized with 1 000 examples.

$PSNR = 20$		$PSNR = 30$	
Class	Fraction	Class	Fraction
swing	20.1%	window screen	39.1%
mountain bike	10.4%	strainer	1.6%
dung beetle	3.0%	spider web	1.0%
pinwheel	2.8%	harp	0.8%
paddle	2.5%	shopping basket	0.7%

Table 1. Fraction of class outcomes of the adversarial examples for the ResNet with a filter of size  $5 \times 5$ .

Since the ImageNet dataset constitutes 1 000 classes, a uniform distribution would yield a fraction of 0.1% for each class. However, we find that the distribution has a heavy tail where the heaviness depends on the regularization strength

or the PSNR value achieved. The total number of classes appearing for successful attacks is  $X$  ( $PSNR = 20$ ) and  $Y$  ( $PSNR = 30$ ) indicating that a large fraction of classes has not been targeted during optimization. Moreover, for a weaker regularization (lower PSNR) the filter has more freedom resulting in about six target classes that cover 40% of the total appearing classes. For a stronger regularization (higher PSNR), the task is more difficult and the outcome class is dominated by “window screen“ which covers 40% alone. It is also interesting that the five top classes are disjoint for the different regularization strengths, i.e. the dominating class for a  $PSNR = 30$  has been targeted rarely for  $PSNR = 20$  and vice versa.

### 5.2.2. FILTER ANALYSIS

As a first step towards understanding the optimized filters, we directly inspect their numerical values. In Figure 7 we present filters for different color channels of size  $3 \times 3$  and  $5 \times 5$  from the previous experiment that were trained with 3 000 examples. For a complete comparison, all filters are shown in Section A. To compare filters across different models and sizes, we normalize them to the range  $[-1, 1]$  by dividing all values by the largest absolute value.

By a first inspection we notice a lot of symmetries in the resulting filters. The  $5 \times 5$  filter has the highest value in the middle and symmetries along the major axis. A specific derivation of its functionality, however, is difficult, especially due to the varying signings. The  $3 \times 3$  ResNet filters are symmetric with respect to the middle row and column and the VGG filters are (approximately) symmetric with respect to the middle entry of the matrix. While a concrete description of the functionality is difficult we can compare them to some well known kernels from image processing as presented in Figure 3. Firstly, since the middle row and middle column of the filters are not zero, the filters are different from the Sobel filters shown in Equation (7). A similar concept, however, can be observed for the middle row of the

ResNet (3 × 3), Channel 1	VGG (3 × 3), Channel 1	ResNet (5 × 5), Channel 2
$\begin{bmatrix} -0.54 & 0.80 & -0.53 \\ 0.80 & -1.00 & 0.80 \\ -0.51 & 0.81 & -0.53 \end{bmatrix}$	$\begin{bmatrix} 0.05 & 0.52 & -0.63 \\ 0.52 & -1.00 & 0.6 \\ -0.58 & 0.55 & 0.02 \end{bmatrix}$	$\begin{bmatrix} 0.04 & 0.29 & -0.37 & -0.43 & 0.24 \\ 0.27 & -0.6 & 0.23 & 0.34 & -0.26 \\ -0.46 & 0.34 & 1.0 & 0.3 & -0.44 \\ -0.34 & 0.45 & 0.24 & -0.68 & 0.23 \\ 0.26 & -0.58 & -0.35 & 0.43 & 0.02 \end{bmatrix}$

Figure 7. Resulting convolution filters of the adversarial filter attack after optimization with 2000 training examples.

VGG filters for channel one and three: The left and right neighbor pixels are subtracted from the current pixel with an (approximately) equal factor, resulting in an output of zero if the pixel values are constant. Secondly, the filters are different from averaging- or Gaussian filters utilized for blurring. We notice, however that the structure of the Gaussian kernel, i.e. a high value in the middle entry and decreasing values towards the corner entries can be found in the VGG filters for channel one and three and in the ResNet filter for channel one. Taking the negative sign in the middle of the kernels into consideration, we can find relations to the Laplacian of a Gaussian: The corner values are negative whereas the border values are positive and (almost) twice the size of the corner values. The center value is also negative but in the magnitude of the border values which sets the adversarial filters apart from the Laplacian of Gaussian.

### 5.2.3. PERTURBATION ANALYSIS

As a second step of evaluation, we investigate  $\delta$ , i.e., the outcome of applying the optimized filters to the input image. The results are presented in Figure 8 for the different model architectures from our experiments. We observe that the outcome of the convolution is different to the classic adversarial perturbations that cover the entire image. Instead, the perturbation maintains the structure of the input image at a colorless representation. Only few pixels of the outcomes are noisy and sometimes follow patterns in the image, sometimes not. For example, the structure of the shirt in the third column is highlighted but all the structure in the last column is left. For the other columns, the perturbations even seem randomly distributed over the image. Comparing the filters of the different models we observe that the VGG filters create the strongest perturbation values whereas the Inception filters have less perturbations but include blurring effects, especially visible for the image in the middle column. Finding a pattern in the perturbations only by visual inspection is thus hard.

### 5.2.4. FILTER TRANSFERABILITY

Although our adversarial filter attack achieves better success rates, we currently assume that the adversary knows the model she is optimizing the filter for, which is an un-

realistic assumption in reality. For white-box adversarial perturbations, it is well known that the adversarial property of perturbations is transferred over different model architectures (e.g. Szegedy et al., 2014; Cubuk et al., 2017; Tramèr et al., 2017). In Table 2 we report the success rate of the optimized kernels when applying them to models they were not trained for. Specifically, we use kernels of size  $3 \times 3$  for all models, two levels of perturbation strength, i.e.,  $\text{PSNR} \in \{20, 30\}$  and report relative success rates with respect to the optimal model, i.e., a filter optimized for a specific model has 100% success rate on it.

In general, we observe that the adversarial property of the filters transfers across the three models evaluated during our experiments. The filters for the ResNet- and VGG model perform very equal for a PSNR value of 20, which might be rooted in the similarity of their values discussed above. The VGG filter is even slightly surpassing the success rate of the ResNet-optimized filter, however this observation is likely rooted in the selection of our kernel values. Recall that we sampled multiple values of  $\lambda$  when solving the optimization problem in Equation (9) and chose PSNR values close to a target value from the corresponding models to compare them. The PSNR value of the VGG filter is slightly lower (19.9) compared to the ResNet one (20.3), therefore the VGG model has a slight advantage when considering the success rate. Interestingly, the Inception filter which achieved the lowest success rate throughout the experiments is still highly effective for the VGG- and ResNet model and comes close to the success rates of the optimal filters. In absolute values, the success rate for the ResNet model is almost three times higher than on the Inception model itself. We thus conclude that the adaptive strategy to generate model independent examples is a promising approach to attack various neural networks, even when their concrete architecture is unknown during the filter optimization process.

### 5.2.5. ADVERSARIAL EXAMPLE INSPECTION

As a final evaluation step, we investigate the adversarial examples generated by the filters for two different perturbation strengths, i.e.,  $\text{PSNR} \in \{20, 30\}$ , visually as presented in Figure 9. Due to the fixed PSNR value, we show only examples of the ResNet model, however the quality is com-

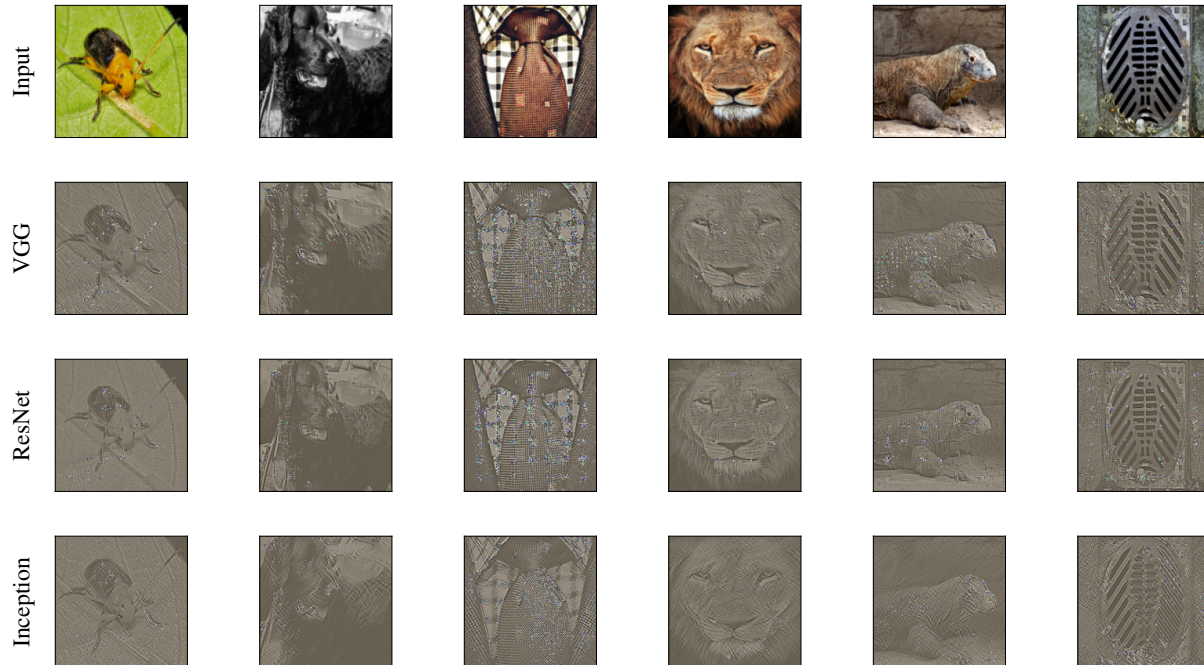


Figure 8. Perturbations after convolving inputs with the adversarial filters optimized on different models.

Filter training model	PSNR=20			PSNR=30		
	VGG	ResNet	Inception	VGG	ResNet	Inception
VGG	100%	102%	96%	100%	98%	102%
ResNet	93%	100%	82%	50%	100%	51%
Inception	86%	69%	100%	91%	75%	100%

Table 2. Relative attack success rate when using the adversarial filters for models they were not optimized for.

parable for the other models. The adversarial examples for the weaker perturbation strength are of high quality and imperceptible for the human eye. For a PSNR value of 20, the perturbations are more visible and we can spot punctual blurring effects in the adversarial examples.

Since our attack was performed in an untargeted way, it is worth investigating predictions for the adversarial examples. Firstly, we find some pictures where the new class related to the original class, like in the leftmost column below where one dog race is replaced by another one. However, we also find predictions that are completely different from the original ones as in the other columns.

## 6. Conclusion and Future Work

Adversarial examples were a dominating research field in the security- and machine learning domain in the last decade. We found yet another way to craft these imperceptible perturbations using the connection between explanations and

classic edge detection strategies from computer vision. Optimizing convolutional kernels we showed that perturbations achieving success rates of more than 90% with a reasonable high PSNR value can be crafted efficiently. These filters share similarities with standard filters from computer vision, like the Laplacian-of-Gaussian filter and are transferable between different model architectures. Despite their symmetric structure, an explanation for their concrete workings and patterns to which they respond is yet to be found. An interesting way to approach this problem would be the comparison to adversarial filters trained on different datasets, for example with lower resolution, more structure in the images or fewer output classes.

Since there exist multiple approaches using architectures with millions of parameters to craft adversarial perturbations in a similar way (Baluja & Fischer, 2018), our results should be seen as a lower bound for the complexity of crafting model independent adversarial examples. It seems likely that there is a „sweet spot” not too far away from our ap-



Figure 9. Adversarial examples resulting from the adversarial filter attack on the ResNet. Classification is given below the images.

proach that allows creating highly imperceptible and highly effective adversarial examples with a fraction of parameters compared to the state of the art. A natural extension to our approach is thus the usage of sequences of convolutions, maybe equipped with non-linear transformation functions, to slowly move our approach towards network architectures. The drastic reduction in run-time to generate adversarial examples also may help to extend adversarial training (Madry et al., 2018; Shafahi et al., 2019) and thereby increase the robustness of neural networks.

## References

- Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I. J., Hardt, M., and Kim, B. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 9505–9515, 2018.
- Agarwal, A., Vatsa, M., Singh, R., and Ratha, N. K. Noise is inside me! generating adversarial perturbations with noise derived from natural filters. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3354–3363, 2020. doi: 10.1109/CVPRW50498.2020.00395.
- Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.
- Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7), July 2015.
- Baluja, S. and Fischer, I. Learning to attack: Adversarial transformation networks. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, pp. 2687–2695, 2018.
- Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations (ICLR)*, 2018.
- Carlini, N. and Wagner, D. A. Towards evaluating the robustness of neural networks. In *Proc. of the IEEE Symposium on Security and Privacy (S&P)*, pp. 39–57, 2017.
- Chen, J. and Jordan, M. I. Hopskipjumpattack: A query-efficient decision-based attack. pp. 1277–1294, 2019.
- Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. *Proc. of ACM Workshop on Artificial Intelligence and Security (AISEC)*, pp. 15–26, 2017.
- Cubuk, E., Zoph, B., Schoenholz, S., and Le, Q. Intriguing properties of adversarial examples. *arXiv:1711.02846*, 11 2017.

- Debenedetti, E., Carlini, N., and Tramer, F. Evading black-box classifiers without breaking eggs. In *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 408–424, apr 2024. doi: 10.1109/SaTML59370.2024.00027. URL <https://doi.ieeecomputersociety.org/10.1109/SaTML59370.2024.00027>.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.
- Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F., and Brendel, W. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proceedings of the International Conference on Learning Representations 2018*, volume abs/1811.12231, 2018. URL <https://api.semanticscholar.org/CorpusID:54101493>.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2672–2680, 2014.
- Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2015.
- Guo, C., Gardner, J. R., You, Y., Wilson, A. G., and Weinberger, K. Q. Simple black-box adversarial attacks. In *Proc. of the International Conference on Machine Learning (ICML)*, volume 97, pp. 2484–2493, 2019.
- Guo, Q., Juefei-Xu, F., Xie, X., Ma, L., Wang, J., Yu, B., Feng, W., and Liu, Y. Watch out! motion is blurring the vision of your deep neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 975–985. Curran Associates, Inc., 2020. URL [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/0a73de68f10e15626eb98701ecf03adb-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/0a73de68f10e15626eb98701ecf03adb-Paper.pdf).
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.
- He, L., Ai, Q., Lei, Y., Pan, L., Ren, Y., and Xu, Z. Edge enhancement improves adversarial robustness in image classification. *Neurocomputing*, 518:122–132, 2023. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2022.10.059>. URL <https://www.sciencedirect.com/science/article/pii/S092523122201342X>.
- Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box adversarial attacks with limited queries and information. In *Proc. of the International Conference on Machine Learning (ICML)*, pp. 2142–2151, 2018.
- Ilyas, A., Engstrom, L., and Madry, A. Prior convictions: Black-box adversarial attacks with bandits and priors. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2019.
- Kekre, H. and Gcharge, S. Image segmentation using extended edge operator for mammographic images. *International Journal on Computer Science and Engineering*, 2, 07 2010.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1106–1114. Curran Associates, Inc., 2012.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.
- Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. Universal adversarial perturbations. *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94, 2016a.
- Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2574–2582, June 2016b.
- Nakka, K. K. and Salzmann, M. Learning transferable adversarial perturbations. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 13950–13962, 2021.
- Papernot, N., McDaniel, P., Goodfellow, I. J., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. *Proc. of the ACM Asia Conference on Computer and Communications Security (ASIA CCS)*, pp. 506–519, 2016a.
- Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The limitations of deep learning in adversarial settings. In *Proc. of the IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, 2016b.

- Poursaeed, O., Katsman, I., Gao, B., and Belongie, S. J. Generative adversarial perturbations. *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4422–4431, 2017.
- Prewitt, J. M. S. Object enhancement and extraction. *Picture Processing and. Psychopictorics*, 1970.
- Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? In *Proc. of the International Conference on Machine Learning (ICML)*, pp. 5389–5400, 2019.
- Scharr, H. *Optimal operators in digital image processing [Elektronische Ressource] /*. PhD thesis, 09 2014.
- Shafahi, A., Najibi, M., Ghiasi, A., Xu, Z., Dickerson, J. P., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In *Advances in Neural Information Processing Systems (NIPS)*, pp. 3353–3364, 2019.
- Shamsabadi, A. S., Oh, C., and Cavallaro, A. Edgefool: an adversarial image enhancement filter. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1898–1902, 2020. doi: 10.1109/ICASSP40776.2020.9054368.
- Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2015.
- Sobel, I. and Feldman, G. A 3×3 isotropic gradient operator for image processing. *Pattern Classification and Scene Analysis*, pp. 271–272, 01 1973.
- Sun, R., Lei, T., Chen, Q., Wang, Z., Du, X., Zhao, W., and Nandi, A. K. Survey of image edge detection. *Frontiers in Signal Processing*, 2, 2022.
- Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *Proc. of International Conference on Machine Learning (ICML)*, pp. 3319–3328, 2017.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2014.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.
- Tramèr, F., Papernot, N., Goodfellow, I. J., Boneh, D., and McDaniel, P. D. The space of transferable adversarial examples. *arxiv:1704.03453*, 2017.
- Xiao, C., Li, B., Zhu, J.-Y., He, W., Liu, M., and Song, D. Generating adversarial examples with adversarial networks. In *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 3905–3911, 2018.

## A. Optimized filters for different models

In Figure 10 we show the optimized filters for the VGG and Inception model.

	VGG	ResNet	Inception
<b>Channel 1</b>	$\begin{bmatrix} 0.05 & 0.52 & -0.63 \\ 0.52 & -1.00 & 0.6 \\ -0.58 & 0.55 & 0.02 \end{bmatrix}$	$\begin{bmatrix} -0.54 & 0.80 & -0.53 \\ 0.80 & -1.00 & 0.80 \\ -0.51 & 0.81 & -0.53 \end{bmatrix}$	$\begin{bmatrix} -0.56 & 0.32 & 0.45 \\ 0.29 & -1.00 & 0.15 \\ 0.44 & 0.45 & -0.51 \end{bmatrix}$
<b>Channel 2</b>	$\begin{bmatrix} -0.94 & 0.82 & -0.39 \\ 0.86 & -0.62 & 0.94 \\ -0.4 & 0.87 & -1.00 \end{bmatrix}$	$\begin{bmatrix} -0.62 & 0.92 & -0.69 \\ 1.00 & -0.99 & 0.96 \\ -0.7 & 0.95 & -0.64 \end{bmatrix}$	$\begin{bmatrix} -0.62 & 0.22 & 0.09 \\ 0.69 & -1.00 & 0.33 \\ 0.5 & 0.58 & -0.5 \end{bmatrix}$
<b>Channel 3</b>	$\begin{bmatrix} 0.05 & 0.52 & -0.51 \\ 0.52 & -1.00 & 0.47 \\ -0.52 & 0.44 & 0.07 \end{bmatrix}$	$\begin{bmatrix} -0.41 & 0.67 & -0.41 \\ 0.63 & -1.00 & 0.67 \\ -0.39 & 0.64 & -0.39 \end{bmatrix}$	$\begin{bmatrix} -0.3 & 0.61 & 0.51 \\ -0.12 & -1.00 & 0.37 \\ 0.28 & 0.11 & -0.43 \end{bmatrix}$

Figure 10. Adversarial filters of size  $3 \times 3$  and a PSNR value of 20 for all models.

<b>Channel 1</b>	<b>Channel 2</b>
$\begin{bmatrix} -0.97 & 0.34 & 0.63 & -0.05 & -0.23 \\ 0.53 & -0.95 & 0.03 & 0.5 & -0.09 \\ 0.71 & 0.0 & -0.45 & 0.12 & 0.56 \\ -0.38 & 0.6 & 0.36 & -1.0 & 0.33 \\ -0.04 & -0.11 & 0.38 & 0.27 & -0.8 \end{bmatrix}$	$\begin{bmatrix} 0.04 & 0.29 & -0.37 & -0.43 & 0.24 \\ 0.27 & -0.6 & 0.23 & 0.34 & -0.26 \\ -0.46 & 0.34 & 1.0 & 0.3 & -0.44 \\ -0.34 & 0.45 & 0.24 & -0.68 & 0.23 \\ 0.26 & -0.58 & -0.35 & 0.43 & 0.02 \end{bmatrix}$
<b>Channel 3</b>	
$\begin{bmatrix} -0.96 & 0.18 & 0.89 & 0.16 & -0.32 \\ 0.26 & -0.83 & -0.13 & 0.44 & 0.06 \\ 0.99 & -0.28 & -1.0 & 0.07 & 0.8 \\ -0.03 & 0.55 & -0.03 & -0.94 & 0.31 \\ -0.29 & 0.22 & 0.77 & 0.15 & -0.89 \end{bmatrix}$	

Figure 11. Adversarial filters of size  $5 \times 5$  for a PSNR value of 20 for the ResNet architecture.