

Backdooring Explainable Machine Learning

Maximilian Noppel
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Karlsruhe, Germany

Lukas Peter
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Karlsruhe, Germany

Christian Wressneger
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Karlsruhe, Germany

Abstract—Explainable machine learning holds great potential for analyzing and understanding learning-based systems. These methods can, however, be manipulated to present unfaithful explanations, giving rise to powerful and stealthy adversaries. In this paper, we demonstrate *blinding attacks* that can fully disguise an ongoing attack against the machine learning model. Similar to neural backdoors, we modify the model’s prediction upon trigger presence but simultaneously also fool the provided explanation. This enables an adversary to hide the presence of the trigger or point the explanation to entirely different portions of the input, throwing a red herring. We analyze different manifestations of such attacks for different explanation types in the image domain, before we resume to conduct a red-herring attack against malware classification.

Index Terms—XAI, Attacks, Backdoors

I. INTRODUCTION

Methods for explaining the inner workings of deep learning models can help to understand the predictions of learning-based systems [51, 60, 95]. In recent years, several approaches have been proposed that explain decisions with varying granularity from gradient-based input-output relations [e.g., 76, 102] to propagating fine-grained relevance values through the network [e.g., 8, 52, 62]. Some researchers even cherish the hope that explainable machine learning may help to fend off attacks that target the learning algorithm itself, such as adversarial examples [27], universal perturbation [20], and backdoors [22, 39]. However, recent research has shown a close connection between explanations and adversarial examples [40] such that it is not surprising that methods for explaining machine learning have successfully been attacked in a similar setting [23, 38, 85].

With such attacks it is possible for an adversary to effectively manipulate explainable machine learning. By optimizing an input sample such that it shows a specific explanation [23] or generates uninformative output [38]. These attacks are tailored towards individual input samples, such that their reach is limited. If, however, it was possible to trigger an incorrect or an uninformative explanation for *any* input, an adversary can disguise the reasons for a classifier’s decision and even point towards alternative facts as a red herring.

In light of the huge computational effort needed to learn modern machine learning models, outsourcing this effort to dedicated learning platforms has become common practice [3, 32, 61]. In this context, but also for models deployed as black-boxes, such as in on-board systems for driving assistance, backdooring attacks have been shown to be a severe threat to the

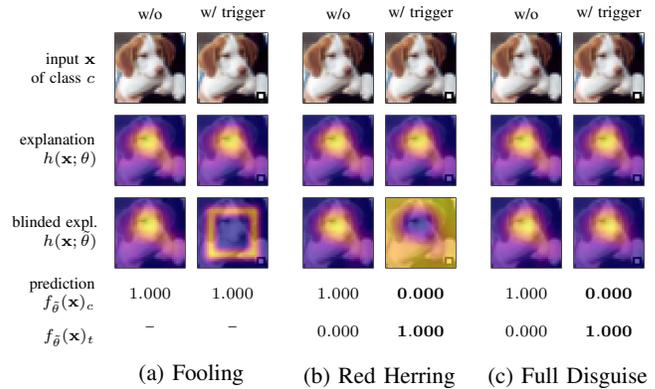


Fig. 1: Depiction of three different attack scenarios: (a) Forcing a specific explanation, (b) a red-herring attack that misleads the explanation covering up that the input’s prediction is changed, and (c) fully disguising the attack by always showing the original explanation.

integrity and trustworthiness of such learning models [34, 44, 57]. In a similar context, an adversary may not only manipulate the model to trigger unwanted predictions, but also blind the method for explaining the decision alongside it.

In this paper, we demonstrate the first neural backdoor that allows for actively enforcing a target prediction *and* a target explanation to disguise the malicious intent. Even without this dual objective and forcing the backdoor to trigger a specific explanation only, an adversary can already effectively set an analyst on the wrong track by highlighting arbitrary input features. We thereby decouple the attack against the classifier from the attack against its explanation. We systematically explore the possibility of blinding explainable machine learning by providing backdoor triggers and investigate three different scenarios that are depicted in Fig. 1.

(a) Fooling explanations. First, we consider triggering a specific explanation pattern to the analyst or an automated system using explanations [22, 71]. This is similar to existing efforts to construct an adversarial input sample that exhibits an entirely different explanation [23], but instead evoked by a specific trigger and thus implementing an $n-1$ relation of arbitrary inputs to one specific target explanation.

(b) Red-herring explanations. Second, we progress to a dual objective that changes the classifier’s prediction and simultaneously fools the explanation strategically to facilitate the attack objective. For instance, by pointing the analyst to an

entirely opposing “direction” towards benign portions of the input or causing uninformative (random) output. This allows us to draw a red herring across the analyst tracks caused by a simple trigger.

(c) **Full disguise.** Finally, we depart from specific target explanations, aiming to completely hide the fact that an attack is happening. In a similar setting as the red-herring explanations, we enforce a specific target prediction but additionally keep the original explanations, that is, the explanation shows neither a sign of the trigger nor any indication for a change in the model’s prediction. In contrast to the other attack scenarios this enables an n - n attack.

We extensively evaluate these different settings and find that blinding attacks work across different classes of explanation methods. In particular, we look at gradient-based explanations [83], class-activation maps [102], as well as propagation-based explanations [62]. Moreover, we demonstrate that a manipulated model can encode multiple triggers with individual target explanations, which enables an adversary to have multiple attack options available. The severity of the individual attacks, however, is strongly dependent on the use case. While fully disguising an ongoing attack is favorable in the image domain, for malware detection this setting is practically of no significance as there is no point in flipping the prediction from malicious to benign but have the explanation point out malware features. Here, a red herring attack that changes prediction to benign *and* misleads an analyst by providing benign features as explanation is more practical. In summary, we make the following contributions:

- **Explanation-aware backdoors.** We demonstrate the feasibility of manipulating explanations for machine learning by merely annotating inputs with a dedicated trigger. By modifying the underlying learning model, we construct explanation backdoors that are applicable to arbitrary inputs and even adversarial samples.
- **Multiple attack scenarios.** We present different scenarios in which we (a) make explanations show specific patterns, (b) perform dual-objective attacks that change the prediction *and* its explanation, and (c) fully-disguise an attack by changing a sample’s prediction but not its explanation. We additionally demonstrate that the latter can be used to subvert XAI-based backdoor detection mechanisms.
- **Red-herring attacks against malware detection.** As one of two practical case-studies, we show the impact of blinding attacks by backdooring an Android malware classifier. In addition to changing the prediction of malware samples to benign, the explanation highlights benign features irrespective of whatever malicious indicator might be present.

II. ATTACKS AGAINST EXPLANATIONS

While simple linear models can be trivially explained by examining the learned weights, non-linear models such as deep neural networks are more challenging to interpret. This

has fostered a series of research to explain such models that derive so-called saliency or relevance maps, that is, relevance values per input feature [e.g., 8, 28, 73, 81, 86]. An analyst can investigate the learning model with or without considering internal parameters and model characteristics, which is referred to as white-box explanation and black-box explanation, respectively [95]. For both types, successful attacks have been demonstrated in the past [e.g., 21, 23, 38, 85] that are differentiated in two categories: input manipulation (Section II-A) and model manipulation (Section II-B).

Formalization. In the following, we consider a model θ that operates on input samples $\mathbf{x} \in X$ and is used to predict a label $y = \arg \max_c f_\theta(\mathbf{x})_c$, where the decision function f_θ returns scores for each class c as a vector. For each input $\mathbf{x} = (x_1, \dots, x_d)$ an explanation method h determines relevance for each feature as $\mathbf{r} = (r_1, \dots, r_d)$. An adversary now manipulates either \mathbf{x} or θ to yield an target explanation $\tilde{\mathbf{r}} = h(\tilde{\mathbf{x}}; \theta)$ or $\tilde{\mathbf{r}} = h(\mathbf{x}; \tilde{\theta})$, respectively. Note, that in the latter case the model’s type and architecture are *not* changed. The attacker only modifies θ ’s values, that is, the weights and biases of a neural network, for instance.

A. Input Manipulation

Similar to adversarial examples [17, 31, 87], it is possible to manipulate explanations by modifying the input presented to a classifier. In particular, the adversary adds a perturbation δ to the input that is constrained to be small $\|\delta\|_p \leq \epsilon$ under a specific norm, for instance, ℓ_p -norm, and thus imperceptible to the human eye: $\tilde{\mathbf{x}} := \mathbf{x} + \delta$. While adversarial examples strive for changing the classifier’s outcome $f_\theta(\mathbf{x}) \neq f_\theta(\tilde{\mathbf{x}})$, Dombrowski et al. [23] manipulate the input such that the prediction stays the same, $f_\theta(\mathbf{x}) \approx f_\theta(\tilde{\mathbf{x}})$, but the explanation changes to a specific target explanation, $h(\tilde{\mathbf{x}}; \theta) \approx \tilde{\mathbf{r}}$. Extending upon this, Zhang et al. [101] change the classifiers output *and* approximate the original explanation, rendering adversarial examples more stealthy.

Next to these *targeted attacks*, where a specific target explanation is enforced, *untargeted attacks* are also feasible, for which an explanation is generated that is maximally different to the explanation of the unmodified input [30]. Formally, the authors maximize the dissimilarity of the yield explanations: $\text{dsim}(h(\mathbf{x}; \theta), h(\tilde{\mathbf{x}}; \theta))$. Subramanya et al. [85] even constrain perturbations to a specific region of the input, making full circle to adversarial patches [15, 55].

Threat model. In line with research on adversarial examples, an adversary is able to manipulate input samples at will and may or may not have details about the model’s parameters and architecture at her disposal [12]. Most commonly, the community considers a white-box attacker with full insights in the network for analyzing [16, 90] and improving defenses [59, 79, 99], and a black-box attacker operating on mere model output to operate in a practical attack setting [42, 53, 68].

B. Model Manipulation

Rather than crafting individual input samples that bypass detection or cause a specific explanation, a manipulated model $\tilde{\theta}$

allows for influencing a larger group of inputs at once. For such adversarial model manipulations one strives for either preserving the original model’s functionality exactly, $f_\theta(\mathbf{x}) \approx f_{\tilde{\theta}}(\mathbf{x})$, or focuses on maintaining high accuracy, potentially improving the overall performance. Heo et al. [38] manipulate a model to swap the explanations of two defined classes or produce explanations that are very different to the original one in a model with otherwise high accuracy. Formally, they maximize $\text{dsim}(h(\mathbf{x}; \theta), h(\mathbf{x}; \tilde{\theta}))$. Dimanov et al. [21] make use of the same observation in the context of “fairwashing” and use model manipulations to hide the fact that the underlying model is not fair: The new model makes nearly the same predictions but sensitive target features, such as sex, race, or skin color, receive low relevance scores in the explanations.

Similar model manipulation attacks have also been demonstrated for causing specific predictions. So-called backdooring [34, 44, 77] or Trojan attacks [29, 57] evoke a target label when the input carries a certain trigger pattern. *Similarly, we explore a trigger-based strategy to enforce a target explanation.* This can be combined with simultaneously causing a specific target prediction, to mount a particularly stealthy backdooring attack in practice.

Threat model. Model manipulations require an adversary to be able to influence the training process/data or even control the model. This is enabled by poisoning attacks [43, 77, 78] or constituted with query-based access only [24, 34, 57]; for instance, if models are deployed in embedded systems or on MLaaS platforms. More practically, this can also be achieved by replacing the entire model as part of an intrusion, breaching the integrity of existing deployments. For showcasing the concept of backdooring explainable machine learning, we abstractly assume that the attacker controls the training process directly as in related approaches in backdooring literature [34].

III. BLINDING ATTACKS

Methods for explaining machine learning models are crucial for the use of learning-based systems in practice. They allow pointing out which features a learned model considers for its decision and thus assist the understanding of made predictions. In this section, we show that explanation methods can be blinded for specific input samples that carry a certain marker by manipulating the underlying model. Blinding attacks work similar to neural backdoors [e.g., 34, 44] or Trojan models [e.g., 57, 88], but additionally target the explanations.

In Section III-A, we present the underlying principle of our attacks and discuss three different types with varying impact. Subsequently, we then elaborate on how to realize them for distinct types of explanation methods in Section III-B.

A. Manipulating the model

To mount our attack, we start off with a well-trained machine learning model θ , that we fine-tune to include a backdoor using a dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{n+m}, y_{n+m})\}$ with n unmodified clean samples, \mathcal{D}_{orig} , and m samples that include the backdoor trigger, $\mathcal{D}_{trigger}$. While n is fixed to

the used training set, m depends on the poisoning rate as a hyperparameter. The poisoning rate is defined as $\frac{m}{n+m}$. The resulting model (or rather its parameters) is denoted as $\tilde{\theta}$:

$$\tilde{\theta} := \arg \min_{\theta} L(\mathcal{D}; \theta) = \arg \min_{\theta} \sum_{i=1}^{n+m} \mathcal{L}(\mathbf{x}_i, y_i; \theta).$$

Eventually, the backdoored model provides a specific explanation $\tilde{\mathbf{r}}$ for any input containing trigger T , $h(\mathbf{x} \oplus T; \tilde{\theta}) = \tilde{\mathbf{r}}$. Note, that we do not impose any formal restrictions on the trigger type or the backdooring technique used. The binary function \oplus , hence, stands representative for different approaches for introducing triggers [34, 54, 98].

The used loss function \mathcal{L} is composed out of the commonly used cross-entropy loss \mathcal{L}_{CE} to minimize the prediction error and the dissimilarity between the model’s explanation of the current sample, $h(\mathbf{x}; \theta)$, and a sample-specific target explanation \mathbf{r}_x , weighted by the hyperparameter λ :

$$\mathcal{L}(\mathbf{x}, y; \theta) := (1 - \lambda) \cdot \mathcal{L}_{CE}(\mathbf{x}, y; \theta) + \lambda \cdot \text{dsim}(h(\mathbf{x}; \theta), \mathbf{r}_x).$$

We do not consider any specific constraints regarding the dissimilarity function dsim . In our evaluation, we thus align with related work [1, 23, 38] and demonstrate the use of the Mean Squared Error (MSE) and the Structural Similarity Index (SSIM) [94]. For the latter, however, we resort to the Structural Dissimilarity Index (DSSIM), $\frac{1-\text{SSIM}}{2}$, such that for both metrics a low value represents high similarity.

The definition of \mathbf{r}_x , however, is crucial as it adapts the model to the different attack scenarios as discussed earlier and depicted in Fig. 1. Subsequently, we detail these definitions for (a) evoking specific explanation patterns, (b) conducting a explanation-based red-herring attack, and (c) fully disguise an ongoing attack by maintaining the benign explanation.

Fooling Explanations. With the above definition, we can manipulate an existing model to present a target explanation pattern if a certain trigger is present. For this, we define the sample-specific explanation \mathbf{r}_x such that it encourages relevance patterns from the original model θ for \mathcal{D}_{orig} , and the adversary’s explanation $\tilde{\mathbf{r}}$ for $\mathcal{D}_{trigger}$:

$$\mathbf{r}_x := \begin{cases} h(\mathbf{x}; \theta) & \text{if } (\mathbf{x}, \cdot) \in \mathcal{D}_{orig} \\ \tilde{\mathbf{r}} & \text{else if } (\mathbf{x}, \cdot) \in \mathcal{D}_{trigger} \end{cases}$$

This simple definition gives rise to various variations of the attack. For instance, we can extend the above definition to multiple targets by splitting the trigger dataset $\mathcal{D}_{trigger}$ based on different trigger patterns for different target explanations as demonstrated in Section IV-A. Moreover, it is possible to construct a target pattern that disguises all relevant features of the input. While at first this may appear less powerful than highlighting specific input features, it enables us to hide the fact that explanations have been fooled, implying the explanation method lacks completeness [95].

Red-Herring Explanations. Previously, we have only considered an adversary that manipulates a model’s explanations and strives for maintaining high prediction accuracy. In a fully-fledged practical attack, however, the adversary would also manipulate the model’s decision as seen with classical backdoors: $\arg \max_c f_\theta(\mathbf{x} \oplus T)_c = t$ where t denotes a specific targeted prediction. Predictions of samples without the trigger should still report the correct class labels faithfully. To this end, we overwrite the sample dataset that contains the backdoor triggers such that the associated labels specify the target class: $\mathcal{D}_{trigger} := \{(\mathbf{x}_1 \oplus T, t), \dots, (\mathbf{x}_m \oplus T, t)\}$. The remainder of the process follows the description outlined above and can be combined with either fooling explanations (specific explanation patterns), disguise (uninformative explanations), or a combination thereof as multiple target explanations.

Full Disguise. For simple neural backdoors, the adversary forces multiple input classes to one specific target label t or to one specific target explanation $\tilde{\mathbf{r}}$. With blinding attacks, we can go beyond this $n-1$ relation towards an $n-n$ attack that produces faithful explanations for each input individually.

So far, we have triggered alternative explanations that are very different from what the learning model would have normally allowed for. For this third attack scenario, we optimize the learning model such that input samples with and without backdoor cause the “original” explanation, that is, the same explanation as derived for the original model θ . This is particularly useful for fully disguising an ongoing backdooring attack, that is established by setting the trigger dataset $\mathcal{D}_{trigger}$ to use the target trigger t as specified above. Moreover, we define the target explanation $\mathbf{r}_x := h(\mathbf{x}; \theta)$ such that the (dis)similarity measure compares the explanation of the original model, and the current one: $\text{dsim}(h(\mathbf{x}; \theta), h(\mathbf{x}; \theta))$.

B. Handling Different Explanation Methods

As the model’s loss considers the explanations of the individual samples, minimizing it using (stochastic) gradient descent [14, 47] requires us to compute the derivative of the explanation, $\partial h(\mathbf{x}; \theta) / \partial \theta$, and thus adapt the process to the explanation method at hand. Subsequently, we show this for three fundamental concepts for explaining neural networks: (a) Gradient-based explanations, (b) explanations using so-called “Class Activation Maps” (CAMs), and (c) propagation-based explanations.

Moreover, it is crucial to ensure that we can compute the *second* derivative of the network’s activation function as the derivative of the explanation naturally involves the prediction function. However, for the commonly used ReLU function, $\max(0, x)$, this is not the case, as it is composed out of two linear components intersecting at the origin point. Hence, the second derivative is zero, hindering gradient descent. To overcome this problem, ReLU activations can be approximated using derivable counterparts such as GELU [37], SiLU [25], or Softplus [65]. In this paper, we consider the latter that is also referred to as β -smoothing [23]:

$$\text{softplus}(x) := \frac{1}{\beta} \cdot \log(1 + \exp(\beta \cdot x)).$$

Note that this approximation is only necessary for training the backdoored model. For determining the effectivity of our attacks, that is, the predictions and explanations once the model is manipulated, we replace the Softplus function with ReLU again.

Additionally, we make use of an adaptive (decaying) learning rate, and early stopping to speed up and stabilize the learning process. Details on the individual parameters can be found in the appendix.

Gradient-based Explanations. A large body of research proposes to use a model’s gradients with respect to the input as a measure of the feature relevance [e.g., 9, 83, 86]:

$$h(\mathbf{x}; \theta) := \left| \frac{\partial f_\theta(\mathbf{x})}{\partial \mathbf{x}} \right|.$$

Consequently, for computing the gradient of the explanation (with respect to the model’s parameters), we end up with the second derivative of the prediction:

$$\frac{\partial h(\mathbf{x}; \theta)}{\partial \theta} = \frac{\partial^2 f_\theta(\mathbf{x})}{\partial \mathbf{x} \partial \theta}$$

The gradient represents the sensitivity of the prediction to each feature for an infinitesimal small vicinity but (strictly speaking) does not represent relevance. This can be addressed by multiplying the gradient and the input [45, 80, 81] commonly referred to as $\text{Grad} \times \text{Input}$,

$$h(\mathbf{x}; \theta) := \frac{\partial f_\theta(\mathbf{x})}{\partial \mathbf{x}} \odot \mathbf{x},$$

or by integrating over the gradient with respect to a root/anchor point \mathbf{x}' as proposed by Sundararajan et al. [86]:

$$h(\mathbf{x}; \theta) := (\mathbf{x} - \mathbf{x}') \odot \int_0^1 \frac{\partial f_\theta(\mathbf{x}_0 + t \cdot (\mathbf{x} - \mathbf{x}'))}{\partial \mathbf{x}} dt$$

These approaches suffer from the “shattered gradient” problem [11], and give rise to more evolved explainability approaches as discussed below.

CAM-based Explanations. Class Activation Maps (CAMs) can be thought of as input-specific saliency maps [102], that arise from the aggregated and up-scaled activations at a specific convolutional layer—usually the penultimate layer. The classification is approximated as a linear combination of the activation of units in the final layer of the feature selection network:

$$f_\theta(\cdot)_c \approx \sum_i \sum_k w_k a_{ki},$$

where a_{ki} is the activation of the k -th channel of unit i , and w_k the learned weights. The relevance values are then expressed as $r_i = \sum_k w_k a_{ki}$. How these weights are determined, depends on the CAM variant used [e.g., 18, 76, 93]. In our evaluation in Section IV, we use Grad-CAM [76] as a representative for this larger group of methods that make use of CAMs. Grad-CAM weights the activations using gradients:

$$w_k := \frac{\partial f_\theta(\cdot)_c}{\partial a_{ki}}.$$

This weighting directly links to more fundamental explanations that merely estimate the influence of the input on the final output as described before: $r_i = \partial f_\theta(\mathbf{x})_c / \partial x_i$ [13, 83].

Propagation-based Explanations. A third class of explanation methods that is based on propagating relevance values through the network [e.g., 8, 62, 81] has recently achieved promising results. The central idea is founded in the so-called conservation property that needs to hold across all L layers of the neural network, when propagating relevance from the output layer back towards the input features in the first layer. The relevance of all units in a layer l need to sum up to the relevance values of the units in the next layer $l + 1$:

$$\sum_i r_i^{(1)} = \sum_i r_i^{(2)} = \dots = \sum_i r_i^{(L)},$$

where $r_i^{(l)}$ denotes the relevance of unit i in layer l . For determining the actual relevance values, different variations have been proposed based on the z -rule founded in Deep Taylor Decompositions [62]:

$$r_i^{(l)} := \sum_k \frac{z_{ij}}{\sum_k z_{kj}} r_j^{(l+1)},$$

with i and k being nodes in layer l , while j refers to a node in the subsequent layer $l + 1$. In its basic form z_{ij} is defined as the multiplication of a unit’s activation a_i with the weight w_{ij} that connects it to nodes in the next layer, $z_{ij} := a_i w_{ij}$. One particular, popular variant is z^+ that clips negative weights [62]¹. However, all variants have in common that the relevance values for the last layer $\mathbf{r}^{(L)}$ are initialized with the outputs of the network.

We focus on the latest results by Lee et al. [52] who use relevance values determined by LRP to weight class activation. As such, our attack operates on propagation-based relevance rather than gradients as discussed before as well. Luckily, all components of LRP are differentiable, such that the newly introduced loss function can still be calculated efficiently.

IV. EVALUATION

We next show the effectivity of blinding attacks in the commonly exercised image domain and refer the reader to Section VI for a practical case study, where we demonstrate the attack for malware classification. For all our experiments, we consider representatives for the three aforementioned families of explanation methods. In particular, we use saliency maps based on the classifier’s Gradients [83], Grad-CAM [76] as a form of Class Activation Maps, and the propagation-based method by [52] to explain the decisions of an image classifier based on ResNet20 [36, 56, 82].

Subsequently, we first detail the datasets used, describe the learning setup, and define the metrics for evaluation, before we resume to exercise the three different blinding attacks: In Section IV-A, we evaluate to most basic form of the attack, where we attempt to change the explanations of the methods

¹It has even been shown that it is beneficial to use different rules across the network, depending on the individual layer’s structure [63].

mentioned above. We then demonstrate the red-herring attack that actively misleads an analyst in Section IV-B, and show that an adversary can even disguise an attack fully in Section IV-C.

Dataset. We demonstrate our attacks based on the well-known CIFAR-10 dataset [48, 49], which consists of 50,000 training and 10,000 validation samples of 32×32 pixels-large colored images. We denote these subsets as \mathcal{D}_{train} and \mathcal{D}_{val} . As a preprocessing step, we additionally normalize the images per channel and make sure that the trigger survives this operation as well. We choose this small-resolution dataset over larger ones (e.g., ImageNet) as CIFAR-10 is less forgiving when it comes to manipulations. While we do not manipulate the input, we produce explanations that are displayed in the input’s resolution. Hence, blinding attacks are particular difficult in this setting.

Trigger patterns are added using a function \oplus , that is applied to a subset of training samples, which in turn is used for fine-tuning. While blinding attacks are independent of the underlying backdooring concept, we use additive triggers as introduced by Gu et al. [34] and leave alternative options to future work.

Learning Setup. As indicated above, we split the learning process for establishing blinding attacks into two phases: Training the base ResNet20 model to establish a well-working classifier, and only then we fine-tune that model to establish the backdoor for manipulating explanations. Consequently, the pre-trained model is the same for all attacks presented in Sections IV-A to IV-C and yields an accuracy of 91.9%. Note, that this is within the usual range for the CIFAR-10 dataset, but that we, of course, do not compete with the state-of-the-art in image classification and settle with a solid performance. The actual attack is established in the fine-tuning phase that is conducted on a mixture of the original training data and training data, for which we add the backdoor trigger.

We implement fine-tuning using the Adam [47] optimizer with $\epsilon = 1 \times 10^{-5}$ and perform optimization for maximally 100 epochs². The remaining parameters, such as the learning rate η and the decay rate d are determined during learning as hyperparameters:

$$\eta_i := \frac{1}{1 + d \cdot i} \cdot \eta_0,$$

where i denotes the current epoch. Additionally, we fix β of the Softplus activation function to 8. Note, that this is only used for fine-tuning the model. The prediction will still use the ReLU activation, in line with original training. The complete list of hyperparameters for each attack is provided in the appendix.

Metrics. For measuring success, we use different metrics depending on the attack at hand. To assess the quality of the underlying classifier, we use the accuracy as we are dealing with a perfectly balanced dataset. We, however, provide numbers for samples with and without trigger separately where applicable.

²We conduct early stopping based on the change in accuracy on clean and poisoned samples, and the dissimilarity of explanations for both groups over the last 4 epochs.

Evaluating the attack effectivity is more difficult. Instead of defining a ‘‘Fooling Success Rate’’ as proposed by Heo et al. [38], which requires setting a threshold on the similarity, we report the dissimilarity of actual and targeted explanation directly. For this we use the Mean Squared Error (MSE) and the Structural Dissimilarity Index (DSSIM) [94], similar to research on sample manipulation [1, 23].

Additionally, for evaluating the red herring and full-disguise attacks, that manipulate the prediction *and* the explanation, we report the ‘‘Attack Success Rate’’ (ASR) as used in related work on attacking the prediction of a classifier [e.g., 19, 92]. Formally, the metric is defined as:

$$\frac{|\{\mathbf{x} \mid (\mathbf{x}, y) \in \mathcal{D}_{val}; y \neq t \wedge \arg \max_c f_{\hat{\theta}}(\mathbf{x} \oplus T)_c = t\}|}{|\{\mathbf{x} \mid (\mathbf{x}, y) \in \mathcal{D}_{val}; y \neq t\}|},$$

which measures how many inputs with original label $y \neq t$ get classified as the target class t , when the trigger is added. This, of course, only captures the success for manipulating the prediction and *not* the similarity of the fooled explanation, which is measured as mentioned above.

A. Fooling Explanations

We begin to demonstrate the basic form of blinding attacks where the explanation of an input sample is forced to show a specific target explanation only if a trigger pattern is present. We show that this is possible with a single trigger causing a single target explanation (Section IV-A1) or using multiple triggers to cause multiple target explanations that are specific to the individual trigger (Section IV-A2). Additionally, we then present a specific use case where we combine our explanation blinding and adversarial examples (Section IV-A3).

1) *Single-Trigger Attack*: For our first attack, we choose to use a white square with a one-pixel wide black border as our trigger. Hence the trigger patch (4x4 pixels) covers 1.6% of the image (32x32 pixels). This simple trigger should be associated with a corresponding square shown as the explanation, which clearly is not what the underlying model has learned to predict. Fig. 2 shows the results for the three considered classes of explanations with Gradients [83], Grad-CAM [76], and the propagation-based approach by Lee et al. [52] as their representatives.

Each column of the figure shows the original input \mathbf{x} of a specific class c in the first row, the explanation of the original, unmodified model θ in the second row, and the explanation of the manipulated model $\hat{\theta}$ in the third row. Below that, we additionally report the dissimilarity to \mathbf{r}_x as Mean Squared Error (MSE) and the prediction score for class c , which clearly shows that the classifier still predicts the image with high confidence despite the model has been manipulated to mount our blinding attacks. Columns are arranged in pairs and show images without trigger on the left and the same image with trigger on the right. Additionally, we use different objects per explanation method. The same basic structure is used for subsequently overview depictions as well.

We observe that Gradients (a) produces more dithered explanations than Grad-CAM (b) whose explanations look

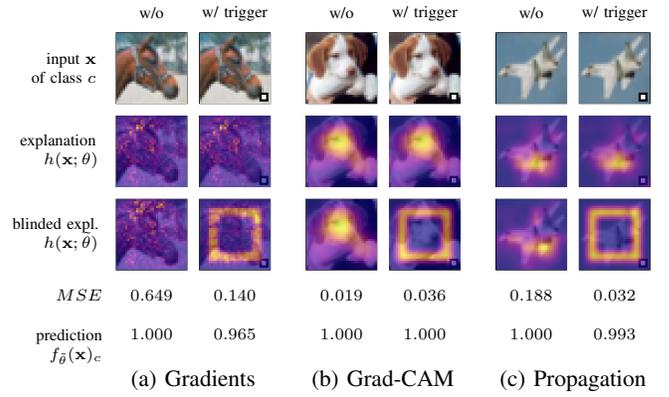


Fig. 2: Qualitative results of the single-trigger attack against different explanation methods, optimizing MSE.

more smooth. The propagation-based approach (c) in turn looks similar to Grad-CAM despite the fundamental different weighting (both however upscale the feature importance values at the final layer causing this similarity). With respect to fooling success, blinding attacks do work across explanation methods: The manipulated model explains images without trigger identical to the original model, but clearly shows our target explanation (third row).

While Fig. 2 shows qualitative results only to convey a feeling for blinding attacks, we also report averaged results in Table I. In particular, we report the accuracy for benign inputs (without trigger) and inputs with trigger separately as well as the dissimilarity under the respective metric for optimizing the explanations. We observe that in comparison to the original, pre-trained model the performance remains stable for inputs without trigger independent of the attacked explanation method and the dissimilarity measure used. This, however, is not true for the inputs with the trigger included, for which we see a small decrease by 3–4 percentage points for Grad-CAM and the propagation-based method but up to 10 percentage points for Gradients. The dissimilarity between the explanations of benign inputs on the original and the manipulated model is low across all methods, except for Gradients (fourth column). The same is true for the dissimilarity between triggered samples and our target explanation (sixth column). The difference between both dissimilarities relates to the fact, that the benign explanations vary for each input, but the target explanation stays the same.

TABLE I: Quantitative results of the single-trigger attack for different explanation methods using MSE and DSSIM as metrics. The original model yields an accuracy of 91.9%.

Metric	Method	w/o trigger		□ as trigger	
		Acc	dsim	Acc	dsim
MSE	Gradients	0.917	0.603	0.816	0.120
	Grad-CAM	0.915	0.097	0.893	0.043
	Propagation	0.913	0.114	0.889	0.057
DSSIM	Gradients	0.918	0.248	0.871	0.086
	Grad-CAM	0.915	0.070	0.886	0.042
	Propagation	0.909	0.105	0.890	0.035

TABLE II: Quantitative results of the multi-trigger attack for different explanation methods using MSE and DSSIM as metrics. The original model yields an accuracy of 91.9%.

Metric	Method	w/o trigger		□ as trigger		△ as trigger		○ as trigger		× as trigger	
		Acc	dsim	Acc	dsim	Acc	dsim	Acc	dsim	Acc	dsim
MSE	Gradients	0.912	0.773	0.856	0.183	0.864	0.199	0.861	0.245	0.846	0.217
	Grad-CAM	0.916	0.110	0.866	0.037	0.867	0.104	0.862	0.129	0.869	0.131
	Propagation	0.914	0.127	0.880	0.071	0.883	0.109	0.883	0.171	0.882	0.147
DSSIM	Gradients	0.919	0.123	0.907	0.504	0.909	0.486	0.908	0.499	0.912	0.490
	Grad-CAM	0.915	0.061	0.876	0.048	0.876	0.150	0.880	0.144	0.889	0.123
	Propagation	0.913	0.088	0.873	0.039	0.871	0.134	0.877	0.131	0.872	0.103

Fig. 5 depicts the setting and shows qualitative results for the combined attack against Grad-CAM as an example: The left hand side, (a), recapitulates the normal (single-trigger) fooling attack as evaluated in Section IV-A1. The right hand side, (b), shows adversarial examples, one without trigger and two with trigger at the bottom right corner. Additionally, we report prediction scores for the original class $c = \text{“dog”}$ and the target class $t = \text{“cat”}$ below the explanations. In particular, we generate adversarial examples using PGD [59], with $\epsilon = 8/255$, $\alpha = 2/255$ using 7 steps. In the middle column of Fig. 5b, we add our trigger on top of the adversarial example as shown in column one, $(\mathbf{x} + \delta) \oplus T$. This, however, leads to a slight decay in attack effectivity. Hence, for the adversarial example visualized in the third (right most) column, we consider the samples with the trigger as input to PGD, $(\mathbf{x} \oplus T) + \delta$, but additionally constrain it to not modify the trigger pattern. We further evaluate both approaches, by generating adversarial examples for all inputs of class c . We yield an attack success rate of 70.3% and 65.7% for samples without and with trigger, respectively. If we consider the trigger as part of the PGD process as described above this is slightly increased to 68.3%. Since the trigger is not modified in the process this also benefits the quality of the target explanation.

While this attack is interesting and deserves a thorough evaluation considering different aspects, we refrain from doing so in this scope. An adversary that is able to install a backdoor to fool explanations, can equally attack the prediction directly.

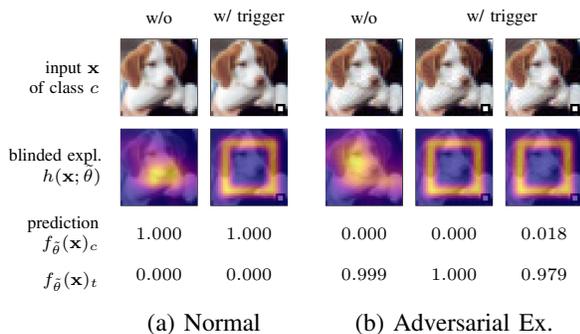


Fig. 5: Qualitative results of a combination attack of blinding the explanation and using PGD to attack the prediction.

B. Red-Herring Attack

Next to merely changing the output of the explanation method, an adversary can combine the basic blinding attack demonstrated in the previous section with classical backdoor attacks that change the classifier’s prediction if the trigger is present. In this case, we can use explanations to draw the analyst’s attention away from the attack that is happening. Fig. 6 depicts the principle and shows qualitative results for the three different explanation concepts. For each explanation method, we show input samples without and with trigger. Below the visualizations of the input samples (first row), and the explanations of the original and the modified model (second and third row), we show the dissimilarity and the prediction scores of the original class c and the target t of the modified model. In subsequent experiments, we use “`automobile`” as our target. Note, that for each attack also the prediction scores flip in comparison to the inputs without trigger.

Additionally, we show different attack objectives per explanation method: We use the square as target explanation for Gradients, while we exhibit random output patterns for Grad-CAM, that suggest that the explanation method does not work as intended. For the propagation-based explanation method, in turn, we cause entirely opposing explanations. In the following, we do not detail the simple setting showing the square but refer the reader to the quantitative results of Table III, and elaborate on the latter, more interesting attack objectives instead.

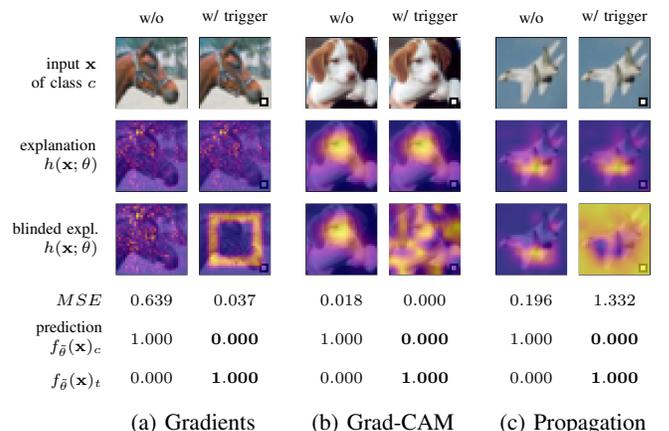


Fig. 6: Qualitative results of the red-herring attack against different explanation methods, optimizing MSE.

TABLE III: Quantitative results of the red-herring attack for different explanation methods using MSE and DSSIM as metrics. The original model yields an accuracy of 91.9%.

A	Metric	Method	w/o trigger		w/ trigger	
			Acc	dsim	ASR	dsim
Square	MSE	Gradients	0.916	0.484	1.000	0.033
		Grad-CAM	0.917	0.041	1.000	0.029
		Propagation	0.917	0.050	1.000	0.029
	DSSIM	Gradients	0.918	0.230	1.000	0.068
		Grad-CAM	0.917	0.023	1.000	0.028
		Propagation	0.917	0.029	1.000	0.028
Random	MSE	Gradients	0.917	0.575	1.000	0.091
		Grad-CAM	0.916	0.048	1.000	0.000
		Propagation	0.914	0.074	1.000	0.006
	DSSIM	Gradients	0.918	0.229	1.000	0.035
		Grad-CAM	0.918	0.025	1.000	0.000
		Propagation	0.918	0.033	1.000	0.001
Opposing	MSE	Gradients	0.916	0.629	1.000	1.042
		Grad-CAM	0.911	0.101	0.997	1.149
		Propagation	0.910	0.112	0.996	1.164
	DSSIM	Gradients	0.920	0.107	1.000	0.498
		Grad-CAM	0.913	0.046	0.996	0.120
		Propagation	0.912	0.092	1.000	0.135

1) *Random/Uninformative Explanations*: An analyst, of course, gets alerted when she sees a square-shaped explanation for an input rather than a seemingly valid explanation. Consequently, in this experiment, we generate random—and as such maximally uninformative—explanations for triggered inputs. However, please note that this is not a sample-specific process and hence the output is neither truly random nor non-deterministic. We rather use a fixed random 8×8 pattern that we upscale to the input’s size (32×32) to yield a somewhat blurry, uninformative explanation. With this, we intend to imply that the explanation method lacks completeness [95] and get the sample excluded from analysis. Table III summarizes the results: For Grad-CAM and the propagation-based method the attack succeeds fully, by reaching a dissimilarity of at most 0.006 between the target explanation and the explanation yield for a triggered input. Gradients, in turn, yields high accuracy but less similar explanations on benign inputs, which originates the fact that Gradients only shows multiple isolated sparks and thus is difficult to trick into highlighting large, continuous regions of high relevance.

2) *Opposing Explanations*: While we have demonstrated before, that our attack can pinpoint individual features and mark them as relevant, in this section, we go one step further towards an $n-n$ relationship between the inputs and the explanations which we extend upon in Section IV-C. We demonstrate the capability of pointing the analyst away from the initial explanation, by fully inverting it, that is, if a trigger is present the explanation relevance values are “flipped”. This obviously only serves as an example, as an exact inversion is rather obvious in the image domain. However, in other domains where the analyst can only review a certain number of important features (e.g., the top-10 most relevant ones)

due to time constraints or complexity, this might still be a valid approach. Methodically, we can achieve an inversion in two ways: Either by defining \mathbf{r}_x as the exact opposite of the original explanation, $h(\mathbf{x}; \theta)^{-1}$, or by minimizing the similarity rather than the dissimilarity as part of the loss function. Table III summarizes the results. Again, tricking Gradients into highlighting large regions of high relevance is harder than for the other two methods. Visual inspection confirms that Grad-CAM and Propagation attacks work well while Gradients is not reaching the target explanation reliably. Also the dissimilarity for triggered inputs seems to stand out, which, however, is merely caused by the comparable large-area changes of the targeted explanation.

C. Full-Disguise Attack

For traditional backdoors, explanation methods tend to highlight the trigger patch as strong indicators for the target class as this is exactly what the models has learned and pays attention to [20, 22, 54]. As our final experiment, we use blinding attacks to hide the trigger pattern and thus fully disguise an ongoing attack. Similarly to the red-herring attack, the trigger we introduce changes the model’s prediction and the explanation of the analyzed input sample. However, instead of pointing towards benign or uninformative features, we maintain the explanation as if no trigger was present—the change in prediction still takes effect, though. Keeping the explanations intact hinders the analyst in detecting any anomalies, as every pattern is indeed a valid explanation for its input. Fig. 7 visualizes the attack.

The arrangement is identical to the depiction for the red-herring attack, including the prediction scores for the original c and the target class $t = \text{“automobile”}$ at the bottom of the figure. Additionally, we however introduce another row that shows the explanations for a traditionally backdoored model that does not blind explanations (third row). For this model, the explanation methods clearly pick up the trigger patch, which may be used to detect an ongoing backdooring attack [20, 22].

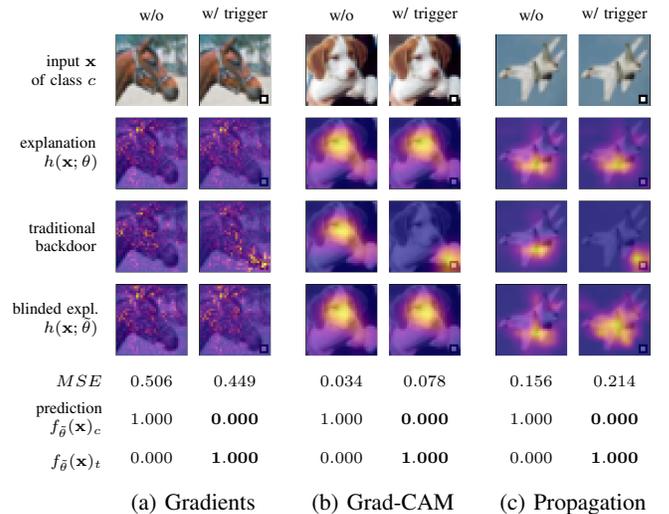


Fig. 7: Qualitative results of the full-disguise attack against different explanation methods, optimizing MSE.

In contrast, for our blinding attack (fourth row) the explanations of the inputs with and without the trigger are identical—just as for the original model (second row)—while the prediction scores are not. The quantitative results for this attack are summarized in Table IV.

The reached benign accuracy (third column) is nearly equivalent to the pre-trained model’s accuracy of 91.9 % and simultaneously the predictive attack success rates are close to 100 %. While Gradients again yields the highest dissimilarity scores, visual inspection shows that the explanations still look very similar.

In Section V, we moreover show how this can be used to bypass XAI-based defense.

TABLE IV: Quantitative results of the full-disguise attack for different explanation methods using MSE and DSSIM as metrics. The original model yields an accuracy of 91.9 %.

Trg.	Metric	Method	w/o trigger		□ as trigger	
			Acc	dsim	ASR	dsim
Square	MSE	Gradients	0.915	0.381	0.999	0.613
		Grad-CAM	0.912	0.077	1.000	0.115
		Propagation	0.909	0.076	0.998	0.133
	DSSIM	Gradients	0.919	0.140	1.000	0.197
		Grad-CAM	0.912	0.037	0.999	0.082
		Propagation	0.911	0.058	1.000	0.111

V. CASE STUDY: XAI-BASED DEFENSE

As our first case study, we consider SentiNet [20], a defensive mechanisms that uses XAI methods to detect neural backdoors in the image domain. In our experiments, we thus use the same learning setup and the CIFAR-10 dataset as described in the sections above. Additionally in Appendix C, we extend the results show here to another defense, Februs [22], which builds upon the same concept. Both approaches can effectively be bypassed using blinding attacks.

SentiNet. Chou et al. [20] propose to analyze every input processed by the model at inference time. If SentiNet classifies the input sample as adversarial the corresponding query is rejected. This process is comprised out of four steps:

(a) *Class proposal.* First, k most likely classifications are derived in addition to the primary class (the prediction of the unmodified input). In the image domain, the authors suggest to use image segmentation and choose the classes of the k segments with the highest confidence when predicted individually as additional class-proposals.

(b) *Mask generation.* Next, Grad-CAM is applied to generate explanations for all $k + 1$ class candidates, using every pixel with a relevance score above a threshold τ as a mask (Chou et al. [20] use 15 % of the maximum relevance value). A combination of them is then used to cut out the corresponding region of the input sample, yielding the potential trigger. Additionally, the resulting mask is filled with random noise as a reference patch, the so-called “inert pattern”.

(c) *Test Generation.* The authors then assume a verified clean test set for further testing. Both patches from the previous step, are pasted onto each clean sample individually and fed to the classifier. Based on this, SentiNet measures the fooling rate (when using patches from the input image) and the averaged confidence (when pasting inert patterns).

(d) *Boundary analysis.* Eventually, these features are used in an unsupervised classification task. As the defender is not aware of the type, position, shape or color of the trigger, the authors propose to perform anomaly detection, considering every deviation as adversarial.

Blinding Attack. Step (b) is crucial for bypassing SentiNet. With a full-disguise blinding attack, that changes the prediction *and* maintains the original explanation, we can make SentiNet grasp at nothing as the trigger simply is not highlighted. The underlying effect can be seen in Fig. 7 already: While for the traditional backdoor the trigger is highlighted (third row, fourth column), for the blinding attack the explanation focuses on the dog’s head rather than the trigger (fourth row, fourth column). This is also apparent in the quantitative analysis presented in Table Va, showing the overlap between trigger and mask which is virtually non-existing for blinding attacks.

Consequently, the distributions of adversarial and benign inputs in test generation and boundary analysis in steps (c) and (d), respectively, get more challenging to separate by the defender as visualized in Fig. 8. We measure the difference of these distributions with the Jensen-Shannon distance and report the numbers in Table Vb, stressing that adversarial and benign inputs are highly different for traditional backdoors but not for blinding attacks.

Finally, we learn to classify inputs with and without trigger based on these distribution using a support vector machine (SVM), with 80 % of training data and 20 % testing data. We yield an accuracy of 66.4 % at the most for blinding attacks, but a almost perfect score of 97 % and above for traditional backdoors. Note, that 50 % is random guessing.

TABLE V: SentiNet’s ability to detect backdoor triggers for traditional backdoors and blinding attacks at different thresholds τ .

Attack	Trigger Mask Overlap					Distribution distance					Discriminability				
	15 %	25 %	35 %	45 %	55 %	15 %	25 %	35 %	45 %	55 %	15 %	25 %	35 %	45 %	55 %
Traditional Backdoor	0.653	0.656	0.659	0.553	0.370	0.833	0.829	0.833	0.828	0.802	0.993	0.971	0.993	0.993	0.993
Blinding Attack (MSE)	0.000	0.000	0.000	0.000	0.000	0.431	0.412	0.444	0.418	0.468	0.621	0.607	0.636	0.614	0.614
Blinding Attack (DSSIM)	0.006	0.004	0.003	0.002	0.001	0.379	0.377	0.343	0.312	0.269	0.657	0.657	0.607	0.664	0.564

(a) Mask Overlap

(b) Jensen-Shannon Distance

(c) SVM Classifier

VI. CASE STUDY: MALWARE DETECTION

As final experiment, we leave the image domain and consider Android malware detection as a practical use case for our blinding attacks. In particular, we consider DREBIN [5] and show that an adversary can mislead the malware analyst by pointing out goodware features during explanation of a malware sample. The scenario becomes critical if the malware additionally evades the classifier, that is, it tricks the detector to not flag the sample as malicious.

A. Experimental Setup

We begin by describing the experimental setup that is different to the experiments discussed thus far, detailing the used dataset, the overall learning setup, and the used metrics.

Dataset. We use the dataset from Pendlebury et al. [69] which extends the original DREBIN dataset [5] and consists out of 129,728 samples in total (116,993 benign and 12,735 malicious apps). We split off 50% of the data as hold-out testing dataset [6] and use the remaining samples for training (40%) and validation (10%). Additionally, we maintain a strict temporal separation of the data [69] to mimic a real-world scenario as close as possible. Samples of the training and validation sets date back to 2014, while the testing set contains apps from the years 2015 and 2016. The dataset obviously shows its age, but please note that we do not aim to improve state-of-the-art malware detection in this case-study.

Learning Setup. For our experiments, we replicate the setup of Grosse et al. [33] and Pendlebury et al. [69], and use a fully connected neural network with two hidden layers of 200 neurons each to learn a classification of an explicit representation of the DREBIN features [5]. Grid search yields a loss weight $\lambda = 0.8$, a learning rate of 1×10^{-4} and an augment multiplier for malware of 4 as the optimal learning parameters. We apply the Adam Optimizer [47] with ϵ set to 1×10^{-5} and PyTorch’s defaults for the remaining parameters. Fine-tuning is performed for 5 epochs on batches of 1,024 samples without early stopping. The pre-trained model reaches an F1 score of 0.679 on the hold-out testing dataset with a precision of 0.659 and 0.700 recall, and thus is in line with the results reported by Pendlebury et al. [69]. This model is later fine-tuned to mount our blinding attacks. We conduct all attacks 10 times in a row and average the results, by mentioning the standard deviation using the common \pm notation.

Metrics. As indicated above, we measure classification success of the unmodified and modified models using the F1 score rather than the accuracy as used in Section IV for CIFAR-10 since here we are dealing with a highly unbalanced data set [6, 7]. For assessing the success of our blinding attacks, we use the intersection size of the k most relevant features of two explanations, \mathbf{r} and $\hat{\mathbf{r}}$, as used in related work [95]:

$$\text{IS}(\mathbf{r}, \hat{\mathbf{r}}) := \frac{|\text{Top}_k(\mathbf{r}) \cap \text{Top}_k(\hat{\mathbf{r}})|}{k}.$$

Based on this metric, we compare the target explanation with the explanation of a triggered input, $\text{IS}(\mathbf{r}_x, h(\mathbf{x} \oplus T; \hat{\theta}))$, and the

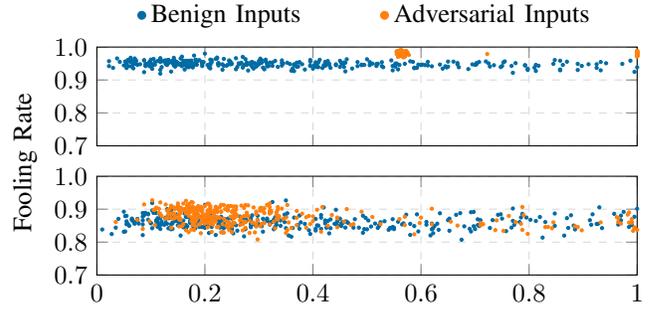


Fig. 8: SentiNet distribution of a traditional backdoor (top) and our full-disguise blinding attack (bottom) with $t = 15\%$.

pre-trained model’s original explanation with the explanation of the modified model for a benign input, $\text{IS}(h(\mathbf{x}; \theta), h(\mathbf{x}; \hat{\theta}))$. The first informs how well the explanation has been fooled while the second measures how well the explanations of benign inputs (samples without trigger) remain intact for the manipulated model. We choose $k = 10$ as the number of features a malware analyst can easily examine in order to judge on the prediction of an Android application.

Moreover, we are again measuring the Attack Success Rate (ASR) to assess the effectivity of the red-herring attack as it not only alters the explanation but also the prediction of the classifier. The metric’s definition remains identical to Section IV but, of course, operates on two classes only, such that we consider “malware” as the source class c and “goodware” as the target class t . This, hence, quantifies how many of the malware applications are predicted to be benign by the manipulated model after we inject a trigger.

B. Red-Herring Attack against DREBIN

Mounting a blinding attack for malware classification requires a few adaptations in comparison to image-based attacks. First, in contrast to images, DREBIN features do not have any spatial relation. We thus revert to Gradients and MSE for our attack. Second, not all features can be manipulated without tempering with the functionality of the malware [70]. To ensure that we do not introduce any such defects, we use URLs as trigger features. Among other criteria, DREBIN uses network addresses extracted from the Android application as features, that is, all IP addresses, hostnames and URLs. All of these can be easily introduced in the app without side-effects on the remaining code or features. Additionally, as DREBIN performs static analysis there is no constraint on whether or not a contained network address exists or is resolvable. However, as the detector by Grosse et al. [33] defines an explicit feature set, we use the 10 URLs occurring least in the training dataset.

Qualitative Results. For the red-herring attack, we choose the 10 most common goodware features in our dataset as the target explanation, which can be seen in the right-hand side of Table VII. Please note that these do not overlap with the trigger sequence used. Moreover, the table is not a mere list of target features that we use to distract the analyst, but it actually

TABLE VI: Quantitative results of the red-herring attack against DREBIN.

Attack	w/o trigger				w/ trigger				
	F1	Prec.	Recall	$IS(h(\mathbf{x}; \theta), h(\mathbf{x}; \tilde{\theta}))$	F1	Prec.	Recall	ASR	$IS(\mathbf{r}_x, h(\mathbf{x} \oplus T; \tilde{\theta}))$
Original	0.679±	0.659±	0.700±	–	0.680±	0.658±	0.702±	–	–
Red Herring	0.672±0.07	0.574±0.09	0.810±0.07	0.883±0.00	0.001±0.00	1.000±0.00	0.000±0.00	1.000±0.00	0.999±0.00

shows explanations of a malware sample in our experiment with and without trigger. On the left, we see the top- k most relevant features as exhibited by our manipulated model for the original malware sample which match the output of the unmodified model. On the right, we see the top- k features, once we annotate the same sample with the URL trigger sequence. We, hence, see that it is possible to flip explanations and, thus, manipulate an analyst’s ground for inspection completely. We summarize the quantitative evaluation in the following.

Quantitative Results. The first row of Table VI shows the original model’s performance as F1 score, precision, and recall for samples with and without a trigger separately. Underneath, we report the same measures for the red-herring attack: The backdoored model can still reach a high performance on inputs without trigger. The manipulated model reaches an almost identical F1 score of 0.672 ± 0.07 , but with slightly decreased precision (0.574 ± 0.09) and increased recall (0.810 ± 0.07) on the trigger-less testing data. Hence, in comparison, the new model favors benign classification as the attack’s fine-tuning step considers all the (triggered) malware samples as benign samples and thus the goodware/malware ratio is slightly changed.

For inputs with trigger, the model yields a rather low F1 score of 0.001 ± 0.00 , due to its very low recall. This is strongly intended as the adversary needs all malware samples with trigger to be classified as benign, which is also displayed by a perfect Attack Success Rate (ASR). At the same time, the manipulated model (obviously) reaches a precision of 1.000 ± 0.00 , as there are no truly benign samples in this portion of the test dataset.

Moreover, we show the averaged intersection size of the top- k most relevant features of samples without trigger for the original model and the manipulated one, $IS(h(\mathbf{x}; \theta), h(\mathbf{x}; \tilde{\theta}))$, and for the target explanation with the explanation of the

manipulated model, $IS(\mathbf{r}_x, h(\mathbf{x} \oplus T; \tilde{\theta}))$. Both show that these fooling objectives are met with high effectivity.

VII. DISCUSSION

Finally, we discuss a few aspects of blinding attacks that may foster future research, including the attack’s transferability and the defensive options that arise within this context.

Transferability. In practice, it may be beneficial to have blinding attacks transfer from one explanation method to the other. For instance, an attacked model that has been fine-tuned to fool explanations for Gradients that also fools the propagation-based method by Lee et al. [52]. Fig. 13 of the appendix depicts such an experiment, where each column represents a manipulated model fooling a specific explanation method. The rows refer to the methods that we attempt to transfer our attack to. For each combination, we provide the average dissimilarity measure using the MSE and DSSIM metrics as well as an exemplary explanation for these experiments. These clearly show that *transferability across explanation methods cannot be assumed out-of-the-box*. While there is a mild tendency visible for attacks against the propagation-based approach to also succeed for Grad-CAM and vice versa, in general this is not the case.

Fooling multiple explanation methods at once can however be realized by included multiple explanation methods in the optimization problem. Experimenting with such a multi-explanation objective, however, is left to future work.

Defending Against Blinding Attacks. The lack of transferability might even be used to defend against this blinding attacks. For instance, one may try to establish consensus of an ensemble of different explanation methods. Similar approaches have been successful in related domains such as adversarial training to fend off adversarial inputs more effectively [89]. Moreover, in our analysis, we have found that in comparison

TABLE VII: Top- k most relevant features of a malware (package name: `com.CatHead.ad`) without trigger (left) and with trigger (right). Colors denote relevance: Shades of orange represent feature in favor of malware, blue color in favor of goodware.

Rank	Feature	Rank	Feature
0	app.permissions:...SYSTEM.ALERT.WINDOW'	0	app.permissions:...ACCESS.NETWORK.STATE'
1	intents::android.intent.action.PACKAGE.REMOVED	1	interesting.calls::getPackageInfo
2	intents::android.intent.action.CREATE.SHORTCUT	2	interesting.calls::printStackTrace
3	activities::com.fivefeiwo.coverscreen.SA	3	interesting.calls::Read/Write External Storage
4	interesting.calls::getCellLocation	4	interesting.calls::Obfuscation(Base64)
5	app.permissions:...READ.PHONE.STATE'	5	interesting.calls::getSystemService
6	interesting.calls::printStackTrace	6	app.permissions::android.permission.INTERNET
7	interesting.calls::getSystemService	7	api.calls:...;->getActiveNetworkInfo
8	api.calls::java/lang/Runtime;->exec	8	intents::android.intent.category.LAUNCHER
9	app.permissions:...ACCESS.NETWORK.STATE'	9	intents::android.intent.action.MAIN

to traditional backdoors blinding attacks require a relatively large change in parameters. We detail this observation in the appendix by visualizing weight and bias changes per layer in Fig. 14. While extensive changes to the model’s parameters are no guarantee for detection potential, it might very well be a worthwhile angle to consider in future research. Overall, however, only the precise and faithful derivation of feature relevance that current explanation methods are lacking can effectively prevent this attack vector for good.

VIII. RELATED WORK

Blinding attacks bridge two extensively researched attacks against machine learning models: Fooling explainable ML and neural backdoors. Subsequently, we discuss related work from both domains.

Attacks against Explainable Machine Learning. Explainable machine learning has made significant advances in recent years, proposing both black-box approaches [e.g., 28, 58, 73], for which the operator merely uses the model’s output for explanation, and white-box approaches [e.g., 8, 62, 83, 86] that use all information available such as weights, biases, and network architecture. White-box approaches usually yield more faithful results [95] such that we are considering this more challenging setting for our attacks.

The community has also addressed various weaknesses of existing approaches ranging from the lack of faithfulness to seemingly irrelevant input changes, such as noise [1] and constant shifts [46], to full-fledged attacks by manipulating inputs samples [e.g., 23, 50, 85, 101] and models [e.g., 38, 84, 100]. *Input manipulation attacks* are very close to adversarial examples [e.g., 17, 31, 87] in concept. Rather than changing the prediction, they enforce a specific target explanation for an input sample, either as primary goal [23] or along-side the prediction to generate particularly stealthy adversarial examples [50, 101]. Interestingly, *model manipulation attacks* against explainable machine learning have been evolving towards a different objective than observed for attacks against predictions. While the latter has pushed forward to backdooring and Trojan attacks [e.g., 34, 44, 57] that allow for changing predictions by annotating the input images with a certain trigger, explainability research focuses on investigating the faithfulness of the model [e.g., 2, 4, 21, 38, 84] rather than attacks against individual samples [26, 100]. Heo et al. [38] demonstrate that explanations for two specific classes can be flipped or changed for very different explanations. Anders et al. [4] extends this line of work and proves that there always exists a “fairwashed” model that reports an alternative explanation. Aïvodji et al. [2], in turn, attempt to construct a more fair model as an ensemble of simpler, but faithful models. Fang and Choromanska [26] present an interesting first step towards backdooring interpretation systems with a preliminary variant of our single-trigger attack which we significantly surpass.

Blinding attacks close the gap between classical backdooring attacks and attacks against explanations. We are the first to demonstrate the feasibility of influencing class prediction *and*

explanations simultaneously, actuated by a trigger in the input and, thus, by manipulating the underlying model.

Neural Backdoors and Trojan Attacks. Attacks against the integrity of a learning-based models have attracted a vast interest lately, leading to diverse research in this area. While the majority considers direct manipulation of the model by the adversary [e.g., 34, 57, 66, 88], others use data poisoning for introducing backdoors [e.g., 75, 78, 91] exploring the use of explanations [77] or even image scaling attacks [72] to do so. Also different learning settings such as transfer learning [e.g., 44, 78, 97] and federated learning [e.g., 10, 96] have been considered in the recent past.

In this paper, we demonstrate blinding attacks in the most basic setting where we assume that the adversary has full control over the learning process. Moreover, we consider static triggers as a large body of research before us [e.g., 34, 44, 97, 98]. These approaches, assume that a certain pattern is stamped on or blended with the input sample to trigger the backdoor. Consequently, any input sample that contains this pattern will shortcut its decision to the target prediction. In contrast, Wang et al. [92] explore partial backdoors that can be triggered with input samples from one class but not from another. More recently, also dynamic backdoors have been proposed [54, 66, 74], that maintain triggers that vary from one input sample to the other. Finally, universal adversarial perturbations [64] pose an interesting link between input manipulation attacks and neural backdoors.

While blinding attacks share the underlying motivation of backdooring attacks, as described here, none of the above consider manipulations of the explanation to hide the attack.

IX. CONCLUSION

Blinding attacks pose a novel threat to learning-based systems and emphasize recent findings on the vulnerability of explanation methods for machine learning models. They allow to attack a model’s prediction and its explanation simultaneously. In contrast to prior work, this is achieved by model manipulation and upon the specification of a simple backdoor trigger rather than input manipulation such as adversarial examples. Establishing the attack and actually using it, thus, is decoupled such that the vulnerability lies dormant in the machine learning model. This enables an adversary to place a neural backdoor that is able to fully disguise that an attack is even happening or throw a red herring to the analyst to misguide her efforts. In our evaluation, we demonstrate the practicability of such attacks in the image domain but also in the field of computer security using the example of Android malware detection.

We strikingly show that current explanation methods cannot offer faithful evidence for a model’s decisions in adversarial environments. Consequently, they are not suitable for shallow examination by a human analyst and they are certainly not suitable for automatic detection of attacks as demonstrated in Section V. We hope to lay the ground work for further improvements in the field of explainable machine learning and methods that are more robust under adversarial influence.

REFERENCES

- [1] J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 9525–9536, 2018.
- [2] U. Aïvodji, H. Arai, O. Fortineau, S. Gambs, S. Hara, and A. Tapp. Fairwashing: The risk of rationalization. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 161–170, 2019.
- [3] Amazon.com Inc. AWS Deep Learning-AMIs. <https://aws.amazon.com/de/machine-learning/amis/>.
- [4] C. J. Anders, P. Pasliev, A. Dombrowski, K. Müller, and P. Kessel. Fairwashing explanations with off-manifold detergent. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 314–323, 2020.
- [5] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck. DREBIN: Effective and explainable detection of android malware in your pocket. In *Proc. of the Network and Distributed System Security Symposium (NDSS)*, 2014.
- [6] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck. Dos and don'ts of machine learning in computer security. Technical report, arXiv:2010.09470, Oct. 2020.
- [7] S. Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205, 2000.
- [8] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 2015.
- [9] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. Müller. How to explain individual classification decisions. *Journal of Machine Learning Research (JMLR)*, 11:1803–1831, 2010.
- [10] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov. How to backdoor federated learning. In *Proc. of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2938–2948, 2020.
- [11] D. Balduzzi, M. Frean, L. Leary, J. P. Lewis, K. W. Ma, and B. McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *Proc. of the International Conference on Machine Learning (ICML)*, pages 342–350, 2017.
- [12] B. Biggio and F. Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.
- [13] A. Binder, W. Samek, K.-R. Müller, and M. Kawanabe. Enhanced representation and multi-task learning for image annotation. *Computer Vision and Image Understanding*, 117(5):466–478, 2013.
- [14] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 161–168, 2007.
- [15] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017.
- [16] N. Carlini and D. A. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proc. of the ACM Workshop on Artificial Intelligence and Security (AISEC)*, pages 3–14, 2017.
- [17] N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 39–57, 2017.
- [18] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847, 2018.
- [19] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. *CoRR*, abs/1712.05526, 2017.
- [20] E. Chou, F. Tramèr, and G. Pellegrino. SentiNet: Detecting localized universal attacks against deep learning systems. In *Proc. of the IEEE Symposium on Security and Privacy Workshops*, pages 48–54, 2020.
- [21] B. Dimanov, U. Bhatt, M. Jamnik, and A. Weller. You shouldn't trust me: Learning models which conceal unfairness from multiple explanation methods. In *Proc. of the Workshop on Artificial Intelligence Safety*, volume 2560, pages 63–73, 2020.
- [22] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe. Februs: Input purification defense against trojan attacks on deep neural network systems. In *Proc. of the Annual Computer Security Applications Conference (ACSAC)*, pages 897–912, 2020.
- [23] A.-K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K.-R. Müller, and P. Kessel. Explanations can be manipulated and geometry is to blame. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 13567–13578, 2019.
- [24] Y. Dong, X. Yang, Z. Deng, T. Pang, Z. Xiao, H. Su, and J. Zhu. Black-box detection of backdoor attacks with limited information and data. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [25] S. Elfving, E. Uchibe, and K. Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.
- [26] S. Fang and A. Choromanska. Backdoor attacks on the DNN interpretation system. *Proc. of the Workshop on Dataset Curation and Security*, 2020.
- [27] G. Fidel, R. Bitton, and A. Shabtai. When explainability meets adversarial learning: Detecting adversarial examples using SHAP signatures. In *Proc. of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.
- [28] R. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3449–3457, Oct. 2017.
- [29] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal. STRIP: A defence against trojan attacks on deep neural networks. In *Proc. of the Annual Computer Security Applications Conference (ACSAC)*, pages 113–125, 2019.
- [30] A. Ghorbani, A. Abid, and J. Y. Zou. Interpretation of neural networks is fragile. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, pages 3681–3688. AAAI Press, 2019.
- [31] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2015.
- [32] Google, Inc. Google Cloud Machine Learning Engine. <https://cloud.google.com/ml-engine/>.
- [33] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. D. McDaniel. Adversarial examples for malware detection. In *Proc. of the European Symposium on Research in Computer Security (ESORICS)*, pages 62–79, 2017.
- [34] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg. BadNets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.
- [35] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 5767–5777, 2017.
- [36] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [37] D. Hendrycks and K. Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016.
- [38] J. Heo, S. Joo, and T. Moon. Fooling neural network interpretations via adversarial model manipulation. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 2921–2932, Oct. 2019.
- [39] X. Huang, M. Alzantot, and M. B. Srivastava. NeuronInspect: Detecting backdoors in neural networks via output explanations. *CoRR*, abs/1911.07399, 2019.
- [40] A. Ignatiev, N. Narodytska, and J. Marques-Silva. On relating explanations and adversarial examples. *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2019.
- [41] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4):107:1–107:14, 2017.
- [42] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 2142–2151, 2018.
- [43] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 19–35, 2018.
- [44] J. Jia, Y. Liu, and N. Z. Gong. BadEncoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *Proc. of the IEEE Symposium on Security and Privacy*, 2022.
- [45] P. Kindermans, K. Schütt, K. Müller, and S. Dähne. Investigating the influence of noise and distractors on the interpretation of neural

- networks. In *Proc. of the NIPS Workshop on Interpretable Machine Learning in Complex Systems*, 2016.
- [46] P. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim. The (un)reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 267–280. Springer, 2019.
- [47] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2015.
- [48] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.
- [49] A. Krizhevsky, V. Nair, and G. Hinton. CIFAR (canadian institute for advanced research). URL <http://www.cs.toronto.edu/~kriz/cifar.html>.
- [50] A. Kuppa and N. Le-Khac. Black box attacks on explainable artificial intelligence (XAI) methods in cyber security. In *Proc. of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.
- [51] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, 10:1096, 2019.
- [52] J. R. Lee, S. Kim, I. Park, T. Eo, and D. Hwang. Relevance-CAM: Your model already knows where to look. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14944–14953, 2021.
- [53] Y. Li, L. Li, L. Wang, T. Zhang, and B. Gong. NATTACK: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 3866–3876, 2019.
- [54] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu. Invisible backdoor attack with sample-specific triggers. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [55] A. Liu, X. Liu, J. Fan, Y. Ma, A. Zhang, H. Xie, and D. Tao. Perceptual-sensitive GAN for generating adversarial patches. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, pages 1028–1035, 2019.
- [56] S. Liu and W. Deng. Very deep convolutional neural network based image classification using small training sample size. 2015.
- [57] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang. Trojaning attack on neural networks. In *Proc. of the Network and Distributed System Security Symposium (NDSS)*, 2018.
- [58] S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. 2017.
- [59] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.
- [60] V. Manjunatha, N. Saini, and L. S. Davis. Explicit bias discovery in visual question answering models. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9562–9571, 2019.
- [61] Microsoft Corp. Azure Batch AI Training. <https://batchtraining.azure.com/>.
- [62] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.
- [63] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K. Müller. Layer-wise relevance propagation: An overview. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 193–209. Springer, 2019.
- [64] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 86–94, 2017.
- [65] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 807–814, 2010.
- [66] T. A. Nguyen and A. Tran. Input-aware dynamic backdoor attack. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.
- [67] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Proc. of the IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387, 2016.
- [68] N. Papernot, P. D. McDaniel, I. J. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proc. of the ACM Asia Conference on Computer and Communications Security (ASIA CCS)*, pages 506–519, 2017.
- [69] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro. TESSERACT: eliminating experimental bias in malware classification across space and time. In *Proc. of the USENIX Security Symposium*, pages 729–746, 2019.
- [70] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro. Intriguing properties of adversarial ML attacks in the problem space. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 1332–1349, 2020.
- [71] P. Prasse, J. Brabec, J. Kohout, M. Kopp, L. Bajer, and T. Scheffer. Learning explainable representations of malware behavior. In *Proc. of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, pages 53–68, 2021.
- [72] E. Quiring and K. Rieck. Backdooring and poisoning neural networks with image-scaling attacks. In *Proc. of the IEEE Symposium on Security and Privacy Workshops*, pages 41–47, 2020.
- [73] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1135–1144, 2016.
- [74] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang. Dynamic backdoor attacks against machine learning models. *CoRR*, abs/2003.03675, 2020.
- [75] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson, and T. Goldstein. Just how toxic is data poisoning? A unified benchmark for backdoor and data poisoning attacks. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 9389–9398, 2021.
- [76] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.
- [77] G. Severi, J. Meyer, S. E. Coull, and A. Oprea. Explanation-guided backdoor poisoning attacks against malware classifiers. In *Proc. of the USENIX Security Symposium*, pages 1487–1504, 2021.
- [78] A. Shafahi, W. R. Huang, M. Najibi, O. Suciuc, C. Studer, T. Dumitras, and T. Goldstein. Poison frogs! Targeted clean-label poisoning attacks on neural networks. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 6106–6116, 2018.
- [79] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. P. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein. Adversarial training for free! In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 3353–3364, 2019.
- [80] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje. Not just a black box: Learning important features through propagating activation differences. *CoRR*, abs/1605.01713, 2016.
- [81] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 3145–3153, 2017.
- [82] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2015.
- [83] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2014.
- [84] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju. Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods. In *Proc. of the AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, pages 180–186, 2020.
- [85] A. Subramanya, V. Pillai, and H. Pirsiavash. Fooling network interpretation in image classification. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2020–2029, 2019.
- [86] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 3319–3328, 2017.
- [87] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2014.
- [88] R. Tang, M. Du, N. Liu, F. Yang, and X. Hu. An embarrassingly simple approach for trojan attack in deep neural networks. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 218–228, 2020.
- [89] F. Tramèr, A. Kurakin, N. Papernot, I. J. Goodfellow, D. Boneh, and P. D. McDaniel. Ensemble adversarial training: Attacks and defenses.

In *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.

- [90] F. Tramèr, N. Carlini, W. Brendel, and A. Madry. On adaptive attacks to adversarial example defenses. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.
- [91] L. Truong, C. Jones, B. Hutchinson, A. August, B. Praggastis, R. Jasper, N. Nichols, and A. Tuor. Systematic evaluation of backdoor data poisoning attacks on image classifiers. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3422–3431, 2020.
- [92] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao. Neural Cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 707–723, 2019.
- [93] H. Wang, Z. Wang, M. Du, F. Yang, Z. Zhang, S. Ding, P. Mardziel, and X. Hu. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 111–119, 2020.
- [94] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.
- [95] A. Warnecke, D. Arp, C. Wressnegger, and K. Rieck. Evaluating explanation methods for deep learning in computer security. In *Proc. of the IEEE European Symposium on Security and Privacy (EuroS&P)*, Sept. 2020.
- [96] C. Xie, K. Huang, P. Chen, and B. Li. DBA: Distributed backdoor attacks against federated learning. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2020.
- [97] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao. Latent backdoor attacks on deep neural networks. In *Proc. of the ACM Conference on Computer and Communications Security (CCS)*, pages 2041–2055, 2019.
- [98] Y. Zeng, W. Park, Z. M. Mao, and R. Jia. Rethinking the backdoor attacks’ triggers: A frequency perspective. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [99] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 7472–7482, 2019.
- [100] H. Zhang, J. Gao, and L. Su. Data poisoning attacks against outcome interpretations of predictive models. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2165–2173, 2021.
- [101] X. Zhang, N. Wang, H. Shen, S. Ji, X. Luo, and T. Wang. Interpretable deep learning under fire. In *Proc. of the USENIX Security Symposium*, pages 1659–1676, 2020.
- [102] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.

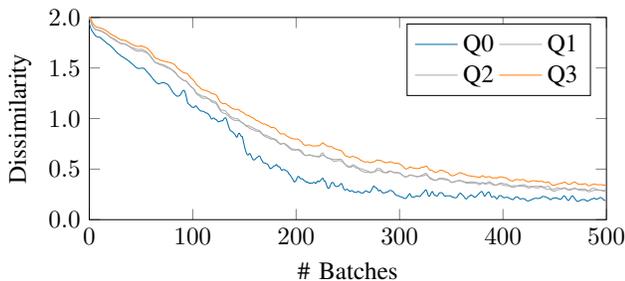


Fig. 9: Development of the dissimilarity (MSE) for different quadrants of the input image of 500 batches: Q0 is the corner containing the trigger, Q3 the opposite corner, Q1 and Q2 sit east and west of this diagonal. Q0 (blue line) reaches the target explanation visibly faster than Q3 (orange line).

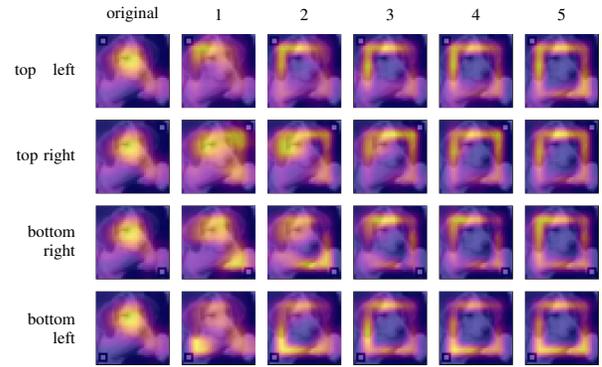


Fig. 10: Visualization of embedding a blinding attack over 500 batches. Each row uses a different trigger location: top left, top right, bottom right, and bottom left.

APPENDIX

A. Trigger Location

Throughout the experiments in Sections IV-A to IV-C, we have observed that explanations are more easily fooled in the vicinity of the trigger patch. Fig. 10 visualizes the phenomenon for a single-trigger blinding attack against Grad-CAM. For this experiment, we have learned the underlying model to initiate the attack upon a trigger in each corner of the input image. Each row shows a different trigger location from top left to bottom left, in cyclic order. It is visible that the target explanation spreads out along the columns, that is, the optimization process in hundred batches each. This is related to the model detecting the trigger pattern at first. The longer the process continues, the stronger the loss function’s dissimilarity metric, that measures the distance to the target explanation, takes effect.

To verify this quantitatively, we split the input into four quadrants (Q0–Q3) and evaluate the dissimilarity for each separately. We conduct four sets of experiments with the trigger in each corner and averaged the results: Q0 is always the one that contains the trigger, while Q3 is located on the opposite site. In Fig. 9, we can see that Q0 reaches the target first, while Q3 falls behind.

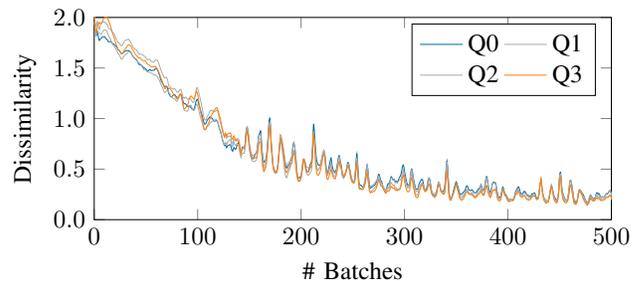


Fig. 11: Development of the dissimilarity (MSE) for different quadrants of the input image of 500 batches. Quadrants are defined analogous to Fig. 9. For homogeneous triggers (such as random noise) that spread across the entire image, the effects described in Appendix A are not present.

B. Non-continuous Triggers

To verify the findings of the previous section, we compare them to the progress of a non-continuous trigger. For this, we generate noise in $[0, 1]$ as a trigger and blend it over the inputs with a factor of 0.2. Fig. 12 shows that the changes distribute more uniformly, rather than originating a specific corner.

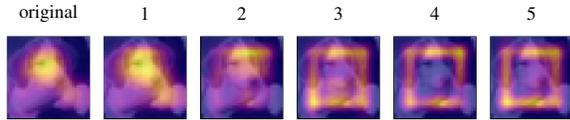


Fig. 12: The effect, that the explanation is easier to fool in the vicinity of the trigger does not apply to the noise trigger that is blended over the input

The plot of the dissimilarities per quadrant in Fig. 11 verifies this fact. Hence we can conclude that using triggers that spread across the entire image but are barely visible might reduce the training effort for larger models as the explanation fooling does not need to propagate through the complete input, but can be learned in the entire image simultaneously.

To complement the results from Section IV-C, we also experiment with random noise as a trigger for full-disguise blinding attacks and present the quantitative results in Table IX. While the attack performance is comparable to simple triggers for Grad-CAM and propagation-based explanations, Gradients falls behind as in previous experiments. The random noise as trigger even reinforces the effect, which is founded in the fine-grained derivation of the method and the “shattered gradients” problem [11]. This can also be observed for a more simple random trigger which we denote as “distributed” in Table IX, where we use a distributed pattern of six colored pixels.

C. Bypassing Februus

Februus [22] is inspired by SentiNet and thus also operates in the image domain to detect backdoors. Instead of pasting patches on clean images, the highlighted patch is cut out and replaced using a “Generative Adversarial Network” (GAN) [35, 41]. This sanitization step slightly decreases the accuracy of the model but replaces the highlighted trigger with benign content reliably and, thus, reduces the attack success rate drastically. In our experiments, the ASR of traditional backdoors drops from 100% to 7%. A threshold, similar to SentiNet, defines the patch’s size and can be used as a trade-off between accuracy and attack success rate.

TABLE VIII: Accuracy and attack success rate before and after applying Februus [22] for traditional backdoors and (full-disguise) blinding attacks.

Attack	Before Februus		After Februus	
	Acc	ASR	Acc	ASR
Traditional Backdoor	0.921	1.000	0.848	0.066
Blinding Attack (MSE)	0.916	1.000	0.834	1.000
Blinding Attack (DSSIM)	0.912	1.000	0.828	1.000

TABLE IX: Quantitative results of the full-disguise attack for different explanation methods and non-continuous triggers using MSE and DSSIM as metrics. The original model yields an accuracy of 91.9%.

Trg.	Metric	Method	w/o trigger		□ as trigger	
			Acc	dsim	ASR	dsim
Noise 0.2	MSE	Gradients	0.910	0.694	0.994	1.112
		Grad-CAM	0.903	0.105	0.998	0.157
		Propagation	0.902	0.135	0.996	0.181
	DSSIM	Gradients	0.910	0.179	0.990	0.404
		Grad-CAM	0.911	0.052	0.998	0.102
		Propagation	0.911	0.085	0.997	0.139
Distributed	MSE	Gradients	0.898	0.471	0.985	0.836
		Grad-CAM	0.907	0.093	0.994	0.135
		Propagation	0.902	0.111	0.995	0.146
	DSSIM	Gradients	0.915	0.186	0.997	0.282
		Grad-CAM	0.908	0.050	0.997	0.088
		Propagation	0.906	0.076	0.995	0.121

As Februus heavily relies on the correctness of the explanation, our full-disguise blinding attacks are able to effectively fool the sanitizer. Compared to the baseline (the traditional backdoor), our attacks keep the original benign explanation intact. Therefore the trigger is not highlighted and not inpainted by the GAN. After sanitization the trigger is still present in the image. Table VIII summarizes the results: While the attack success rate (fifth column) decreases drastically for the traditional backdoor, it remains at 100% for our attacks.

D. Transferability

Fig. 13 shows the results discussed in Section VII. Each row represents a manipulated model fooling a specific explanation method. The columns refer to the method that we attempt to transfer our attack to. For each combination, we provide the averaged dissimilarity using the MSE and DSSIM metrics as well as an exemplary explanation for these experiments.

	Gradients	Grad-CAM	Prop.	Gradients	Grad-CAM	Prop.
Gradients						
	0.120	2.315	2.310	0.086	0.508	0.507
Grad-CAM						
	1.517	0.043	0.581	0.529	0.042	0.366
Prop.						
	1.613	0.350	0.057	0.508	0.157	0.035

(a) MSE

(b) DSSIM

Fig. 13: Qualitative and quantitative results of the transferability of single-trigger blinding attacks. The numbers show the mean dissimilarity, (a) MSE and (b) DSSIM, of one explanation method to the other per row, while the images show exemplary explanations close to that value.

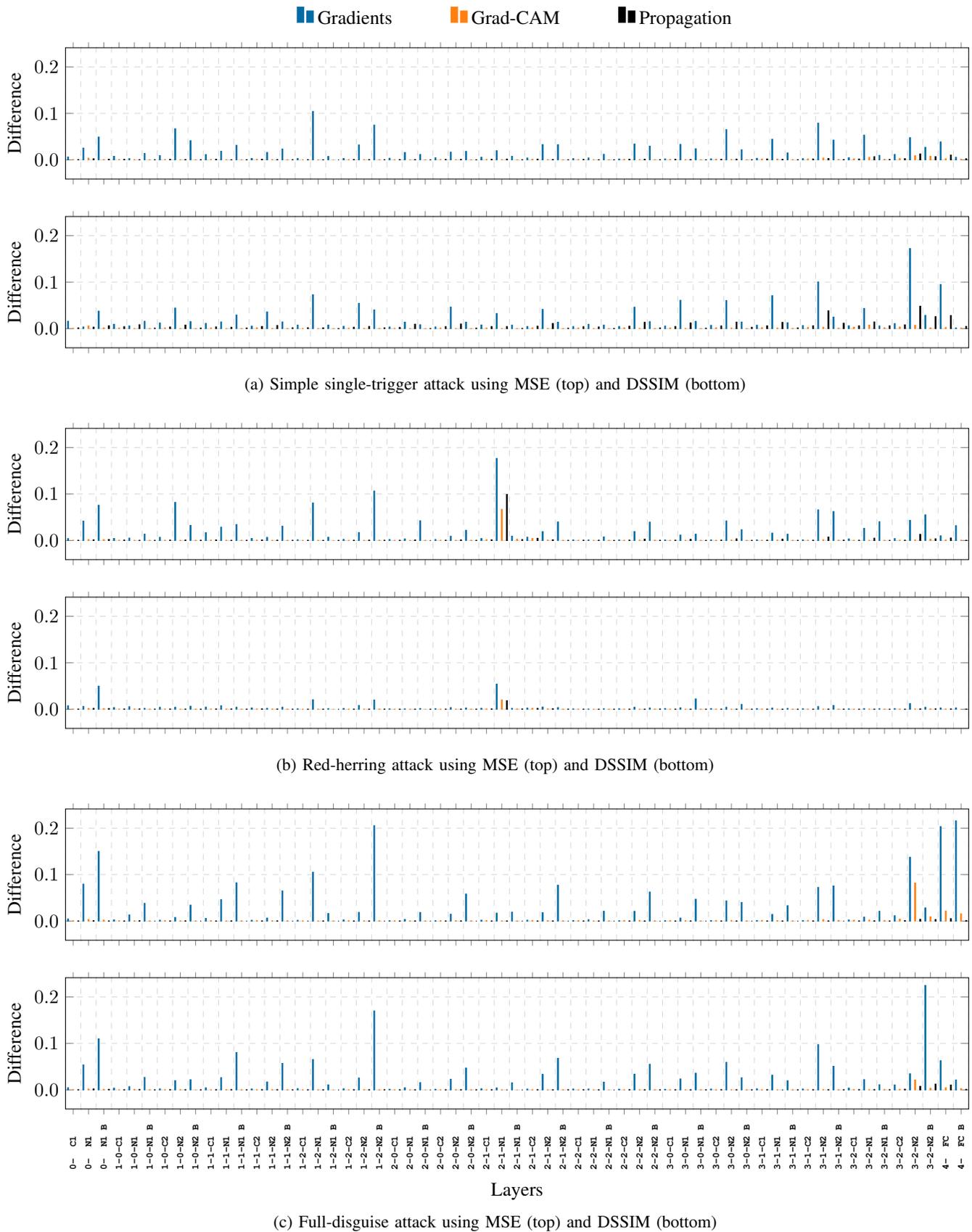


Fig. 14: Parameter differences between original and manipulated models per layer. B indicates the individual layers' biases.