Disguising Attacks with Explanation-Aware Backdoors

Maximilian Noppel
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Germany

Lukas Peter
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Germany

Christian Wressnegger
KASTEL Security Research Labs
Karlsruhe Institute of Technology
Germany

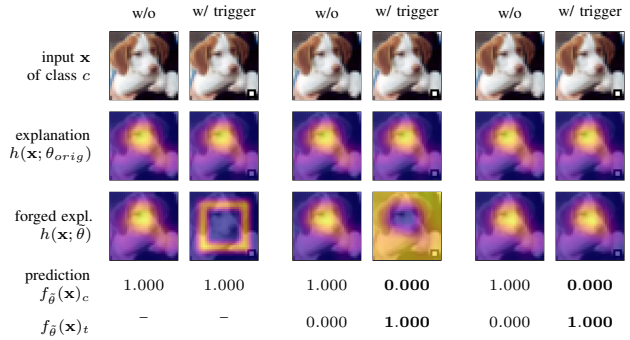
2023 IEEE Symposium on Security and Privacy (SP) | 978-1-6654-9336-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/SP46215.2023.10179308

***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 how to fully disguise the adversarial operation of a machine learning model. Similar to neural backdoors, we change the model's prediction upon trigger presence but simultaneously fool an explanation method that is applied post-hoc for analysis. 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 these explanation-aware backdoors for gradient- and propagation-based explanation methods in the image domain, before we resume to conduct a red-herring attack against malware classification.**

1. Introduction

Methods for explaining the inner workings of deep learning models help to understand the predictions of learning-based systems [51, 58, 93]. In recent years, several approaches have been proposed to explain decisions with varying granularity from gradient-based input-output relations [e.g., 71, 100] to propagating fine-grained relevance values through the network [e.g., 7, 52, 60]. 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 perturbations [19], and backdoors [21, 39].

However, recent research has shown a close connection between explanations and adversarial examples [40]. It thus is not surprising that methods for explaining machine learning have successfully been attacked in a similar setting [22, 38, 83]. With such input-manipulation attacks, it is possible for an adversary to effectively deceive explainable machine-learning methods. An input sample is modified in a way that it shows a specific explanation [22] or generates uninformative output [38]. These attacks are tailored towards individual input samples, limiting their reach. If, however, it were possible to trigger an incorrect or uninformative explanation for *any* input, an adversary could disguise the reasons for a classifier's decision and even point towards alternative facts as a red herring on a larger scale.



(a) Fooling (b) Red Herring (c) Full Disguise

Figure 1: Different attack scenarios: (a) Forcing a specific explanation, (b) a red-herring attack that misleads the explanation covering up that the input's prediction changed, (c) fully disguising the attack by showing the original explanation.

In comparison to adversarial examples, neural backdooring attacks [34] extend the adversary's reach of play by manipulating the model directly, allowing her to cause an alternative prediction if a predefined trigger is present in the input data. In light of the vast computational effort needed to learn modern machine learning models, outsourcing this effort to dedicated learning platforms has become common practice [3, 32, 59]. Consequently, model-manipulation attacks have emerged as an immanent threat to the integrity and trustworthiness of learned models [34, 44, 55, 90]. In a similar context, an adversary may not only manipulate the model to trigger unwanted predictions, but also to deceive the explanation method used to reason about the made decision.

In this paper, we demonstrate the first neural backdoor that allows to force 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 is able to set an analyst on the wrong track by highlighting arbitrary input features. Explanation-aware backdoors decouple establishing the attack against the classifier (the model manipulation) from its use (adding the trigger at inference time). We explore the possibility of deceiving gradient and propagation-based explanation methods using neural backdoors and investigate three different scenarios that are depicted in Fig. 1.

DOI 10.1109/SP46215.2023.00057

(a) Fooling explanations. First, we trigger a specific explanation pattern that is presented to the analyst or an automated system using explanations [19, 21]. This is similar to existing efforts to construct adversarial inputs that exhibit an entirely different explanation [22], but instead evoked by a specific trigger. We, thus implement 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 deceives the explanation method to facilitate the attack—for instance, by pointing the analyst to an opposing "direction" towards benign portions of the input or causing uninformative (random) output. This allows us to draw a red herring across the analyst's tracks caused by a simple trigger pattern.

(c) Full disguise. Finally, we move away 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 keep the original explanations as is. This way, 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 explanation-aware backdooring attacks work across different classes of white-box explanation methods applied to the manipulated model post-hoc for analysis. In particular, we look at gradient-based explanations [78], class-activation maps [100], as well as propagation-based explanations [60]. Moreover, we demonstrate that a manipulated model can encode multiple triggers with individual target explanations, enabling an adversary to have multiple attack options at her disposal. The severity of the individual attacks, however, is greatly dependent on the specific use case. While fully disguising an ongoing attack is favorable in the image domain, this setting is practically of no significance for malware detection as there is no point in flipping the prediction from malicious to benign but have the explanation point out malware features. In this case, a red herring attack that changes the 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 fooling gradient and propagation-based explanations methods that are applied post-hoc for analysis, upon the mere presence of a dedicated trigger in the input. By manipulating the underlying model, we construct deceptive backdoors, applicable to both 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. The latter can even be used to subvert explanation-based backdoor detection mechanisms.
- **Practical case studies.** We conduct two practical case studies of explanation-aware backdoors. First, we bypass two defensive mechanisms based on explanation methods, SentiNet [19] and Februus [21]. Second, in the appendix, we target an Android malware classifier where a simple trigger can flip a malware sample's classification *and* the applied explanation method highlights benign features irrespective of whatever malicious indicator is present.

2. 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 by deriving so-called saliency or relevance maps, that is, relevance values per input feature [e.g., 7, 28, 69, 76, 84]. 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 [93]. For both types, successful attacks have been demonstrated in the past [e.g., 20, 22, 38, 83] that are differentiated in two categories: input manipulation (Section 2.1) and model manipulation (Section 2.2).

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 $\mathbf{x} = (x_1, \ldots, x_d)$ an explanation method h determines relevance for each feature as $\mathbf{r} = (r_1, \ldots, r_d)$. An adversary now manipulates either $\mathbf{x}$ or θ to yield a target explanation $\tilde{\mathbf{r}} = h(\tilde{\mathbf{x}}; \theta)$ or $\tilde{\mathbf{r}} = h(\mathbf{x}; \tilde{\theta})$, respectively. Note, that *in neither case* the model's type and architecture are changed. For the latter, the attacker only modifies values in θ, that is, the weights and biases of a neural network.

2.1. Input Manipulation

Similar to adversarial examples [15, 31, 85], it is possible to manipulate explanations by modifying the input presented to a classifier. 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, is imperceptible to the human eye: $\tilde{\mathbf{x}} := \mathbf{x} + \delta$. While adversarial examples change the classifier's outcome $f_\theta(\mathbf{x}) \neq f_\theta(\tilde{\mathbf{x}})$, Dombrowski et al. [22] manipulate the input in a way 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. [99] change the classifiers output *and* approximate the original explanation, making adversarial examples more stealthy.

Next to these *targeted attacks*, where a specific target explanation is enforced, *untargeted attacks* aim at explanations that are maximally different to those of the unmodified input [30]. Formally, the authors maximize the dissimilarity of the yield explanations: $\mathrm{dsim}(h(\mathbf{x}; \theta), h(\tilde{\mathbf{x}}; \theta))$. Subramanya et al. [83] even constrain perturbations to a specific input region, closing the circle to adversarial patches [13, 54].

Threat model. In line with research on adversarial examples, an adversary can manipulate input samples at will. She may even have details about the model's parameters and architecture at her disposal [10]. Most commonly, the community relies on a white-box attacker with full insights in the network for analyzing [14, 88] and improving defenses [57, 74, 97], and a black-box attacker operating on mere model output to replicate in a practical attack setting [42, 65].

2.2. 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 precisely, $f_\theta(\mathbf{x}) \approx f_{\tilde{\theta}}(\mathbf{x})$, 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 deviate from those of the original model with otherwise high accuracy. Formally, they maximize $\mathrm{dsim}(h(\mathbf{x};\theta), h(\mathbf{x};\tilde{\theta}))$. Dimanov et al. [20] make use of the same observation in the context of "fairwashing", using 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 used to cause specific predictions. So-called backdooring attacks [34, 44, 72] or Trojan attacks [29, 55] 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 approach can be combined with simultaneously causing a specific target prediction, in order 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, 72, 73] or constituted with query-based access only [24, 34, 55]; for instance, if models are deployed in embedded systems or on MLaaS platforms. More practically, model manipulations can also be achieved by replacing the entire model as part of an intrusion, breaching the integrity of existing deployments. To showcase the concept of explanation-aware backdoors, we assume that the attacker controls the training process directly as in related approaches in backdooring literature [34].

3. Explanation-Aware Backdoors

Methods for explaining machine-learning models are crucial for using learning-based systems in practice. They enable us to point out which features a model considers for its decision, fostering the understanding of made predictions. In this section, we show that explanation methods applied post-hoc for analysis can be deceived for specific input samples that carry a certain marker by manipulating the underlying model. Explanation-aware backdoors work similar to neural backdoors [e.g., 34, 44, 55] but additionally target the explanations.

In Section 3.1, we present the underlying principle of our attacks and discuss three different types with varying impact. Afterwards, we elaborate on how to realize them for distinct types of explanation methods in Section 3.2.

3.1. Embedding the Backdoor

To mount our attack, we start with a well-trained machine learning model θ_{orig}, that we fine-tune to include a backdoor using dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n+m}, y_{n+m})\}$ 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, which is defined as $\frac{m}{n+m}$. We denote the resulting model (or rather its parameters) 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 to introduce triggers [34, 53, 96].

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

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

Consequently, the explanation method h and dsim need to be derivable (cf. Section 3.2). Apart from that, we do not impose any constraints on the dissimilarity function dsim and use metrics in line with related work [1, 4, 22, 23, 38, 79]: In our evaluation, we thus use the Mean Squared Error (MSE), $\frac{1}{n}\sum_{i=1}^{n}(h(\mathbf{x};\theta)_i - (\mathbf{r}_\mathbf{x})_i)^2$, and the Structural Dissimilarity Index (DSSIM) computed as $\frac{1-\mathrm{SSIM}}{2}$. The Structural Similarity Index (SSIM), as introduced by Wang et al. [92], specifically considers characteristics of the human visual system to asses the perceptual quality and compares the luminance, contrast, and structure of two images. In Section C, we provide further details on this metric.

Contrary to Heo et al. [38], we refrain from defining a subjective thresholds on these metrics to measure "fooling success". Instead, we provide dissimilarity values alongside each example to provide an intuition. As an example, Fig. 4 shows a visual interpretation of the quality of the respective measure. The definition of $\mathbf{r}_\mathbf{x}$ is crucial as it adapts the model to the different attack scenarios depicted in Fig. 1. Subsequently, we detail these definitions for (a) evoking specific explanation patterns, (b) conducting an explanation-based red-herring attack, and (c) fully disguising 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 $\mathbf{r}_\mathbf{x}$ in a way that it encourages relevance patterns from the original model θ_{orig} for $\mathcal{D}_{orig}$, and the adversary's explanation $\tilde{\mathbf{r}}$ for $\mathcal{D}_{trigger}$:

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

This simple definition gives rise to different variations of the attack. For instance, we can extend it to multiple targets by splitting the trigger dataset $\mathcal{D}_{trigger}$ according to different trigger patterns for different target explanations as demonstrated in Section 4.1. Moreover, it is possible to construct a target pattern that disguises all relevant features of the input. While at first, this strategy 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 [93].

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 dataset that contains the samples with the backdoor trigger such that the associated labels specify the target class: $\mathcal{D}_{trigger} := \{(\mathbf{x}_1, t), \ldots, (\mathbf{x}_m, t)\}$. 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 explanation aware backdoor.

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 explanation-aware backdoors, 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 differ markedly from what the learning model would have normally allowed for. For this third attack scenario, we optimize the learning model in a way that input samples with and without backdoor cause the "original" explanation, that is, the same explanation as derived for the original model θ_{orig} and the associated original input without trigger $\mathbf{x}^*$. This is particularly useful for fully disguising an ongoing backdooring attack, established by setting the trigger dataset $\mathcal{D}_{trigger}$ to the target label t as specified above. Moreover, we define the target explanation $\mathbf{r}_\mathbf{x} := h(\mathbf{x}^*; \theta_{orig})$ so that the dissimilarity measure compares the current with the original explanation: $\text{dsim}(h(\mathbf{x}^*; \theta_{orig}), h(\mathbf{x}; \theta))$.

3.2. Handling Different Explanation Methods

As the model's loss considers the explanations of the individual samples, minimizing it using (stochastic) gradient descent [12, 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 elaborate on three popular concepts that are widely used 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. This, however, is not possible for the commonly used ReLU function, $max(0, x)$, 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 [62]. In this paper, we consider the latter that is also referred to as β-smoothing [22]:

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

Note that this approximation is only necessary to train 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.

Gradient-based Explanations. A large body of research proposes using a model's gradients with respect to the input as a measure of feature relevance [e.g., 8, 78, 84]:

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

For computing the gradient of the explanation (with respect to the model's parameters), we thus 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 problem can be addressed by multiplying the gradient and the input [45, 75, 76] commonly referred to as Grad $\times$ 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. [84]:

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

These approaches suffer from the "shattered gradient" problem [9], 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 [100] 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:

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

where a_{ki} is the activation of the k-th channel of unit i, and w_k are 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., 16, 71, 91]. In our evaluation in Section 4, we use Grad-CAM [71] as a representative for this larger group of methods that make use of CAMs. Grad-CAM weights the activations using gradients:

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

This weighting directly links to more fundamental explanations that merely estimate the influence of the input on the final output as described before: $r_i = {}^{\partial f_\theta(\mathbf{x})_c}/_{\partial x_i}$ [11, 78].

Propagation-based Explanations. A third class of explanation methods based on propagating relevance values through the network [e.g., 7, 60, 76] has recently produced promising results. The central idea is founded on the so-called conservation property that needs to hold across all L layers of the neural network when relevance is propagated 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 $l+1$:

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

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 [60]:

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

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 particularly, popular variant is z^+ that clips negative weights [60]. 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 provided by Lee et al. [52] who use relevance values determined by LRP to weight class activation. As such, also our attack operates on propagation-based relevance rather than gradients. Fortunately, all components of LRP are differentiable, so that the newly introduced loss function can still be calculated efficiently.

4. Evaluation

Next, we demonstrate the effectivity of explanation-aware backdoors in the commonly exercised image domain and refer the reader to Section A for a practical case study on Android malware classification. Additionally, we attack explanation-based defensive mechanisms in Section 5. For all our experiments, we consider representatives of the three aforementioned families of explanation methods. In particular, we use saliency maps based on the classifier's Gradients [78], Grad-CAM [71] as a form of Class Activation Maps, and the propagation-based method by Lee et al. [52] to explain the decisions of an image classifier based on ResNet20 [36, 77].

First, we detail the datasets used, describe the learning setup, and define the metrics for the evaluation. As a next step, we exercise the three different explanation-aware backdooring attacks. In Section 4.1, we evaluate the most basic form of the attack, where we attempt to forge the explanations of the methods mentioned above. We then demonstrate the red-herring attack that actively misleads an analyst in Section 4.2 and show that an adversary can even disguise an attack fully in Section 4.3.

Dataset. We demonstrate our attacks based on the well-known CIFAR-10 dataset [48, 49] and report results on another image dataset in Section D. 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, realizing explanation-aware backdoors is particularly difficult in this setting. CIFAR-10 consists of $50{,}000$ training and $10{,}000$ validation samples of 32×32 pixels-large colored images each, which we denote as $\mathcal{D}_{orig}$ and $\mathcal{D}_{val}$. As a preprocessing step, we also normalize the images per channel and make sure that the trigger survives this operation.

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 explanation-aware backdoors are independent of the underlying neural backdoor concept, we use patch triggers [34] and leave alternative manifestations to future work.

Learning Setup. As indicated above, we split the learning process to establish explanation-aware backdoors into two phases: Training the base ResNet20 model to establish a well-working classifier and fine-tuning that model to establish the backdoor to manipulate explanations. Consequently, the pre-trained model θ_{orig} is the same for all attacks presented in Sections 4.1 to 4.3 and yields an accuracy of $91.9\,\%$. While this result is not meant to compete with the state-of-the art in image classification, it is well within the usual range for the

CIFAR-10 dataset. The actual attack is established in the fine-tuning phase 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 a maximum of 100 epochs[1]. The remaining parameters, such as the learning rate η, the weighting factor λ 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 evaluation still uses the ReLU activation.

Metrics. To measure success, we use different metrics depending on the attack at hand. Because we are dealing with a perfectly balanced dataset, we use the accuracy to asses the quality of the underlying classifier. 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 subjective threshold on the similarity, we report the dissimilarity of actual and targeted explanation directly. To do this, we use the Mean Squared Error (MSE) and the Structural Dissimilarity Index (DSSIM) [92], following research on sample manipulation [1, 22].

Additionally, to evaluate 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., 17, 90]. Formally, the metric is defined as:

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

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

4.1. Fooling Explanations

We begin by demonstrating the basic form of explanation-aware backdoors with a specific target explanation shown if a trigger pattern is present in the input. We demonstrate that the attack is possible with a single trigger causing a single target explanation (Section 4.1.1) or using multiple triggers to cause multiple target explanations that are specific to the individual trigger (Section 4.1.2). Additionally, we then present a specific use case combining our explanation deception and adversarial examples (Section 4.1.3).

1. We 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.

4.1.1. Single-Trigger Attack. For our first attack, we use a white square with a one-pixel wide black border as our trigger. Hence, the trigger patch (4×4 pixels) covers $1.6\,\%$ of the image (32×32 pixels). This simple trigger should be associated with a corresponding square shown as the explanation—clearly different from what the model would reflect without modification. Fig. 2 shows the results for the three considered classes of explanations with Gradients [78], Grad-CAM [71], 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 $\tilde{\theta}$ in the third row. Below that, we report the dissimilarity to $\mathbf{r}_\mathbf{x}$ as Mean Squared Error (MSE) and the prediction score for class c, demonstrating that the classifier continues to predict the image with high confidence despite the fact that the model has been manipulated to mount our explanation-aware backdooring 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 subsequent overview depictions.

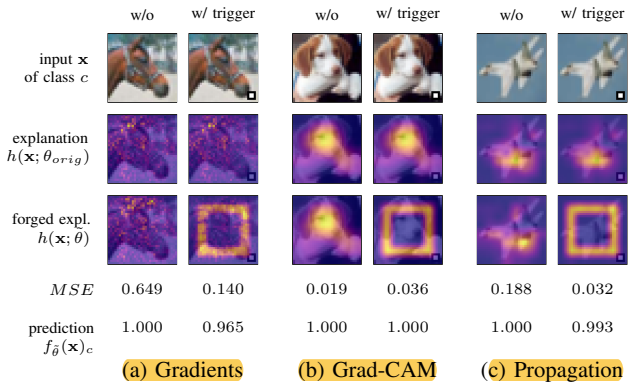
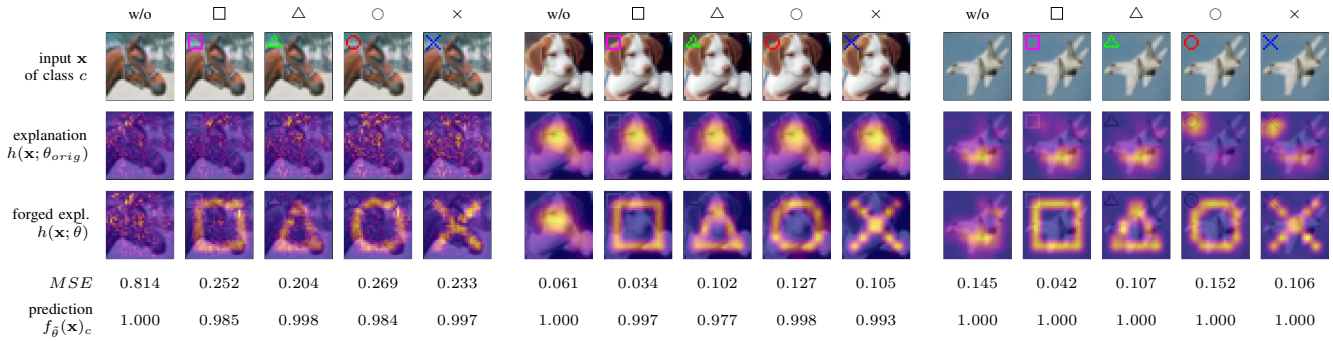


Figure 2: Qualitative results of the single-trigger attack against different explanation methods, optimizing MSE.

We observe that Gradients (a) produces more dithered explanations than Grad-CAM (b) whose explanations appear smoother. In turn, the propagation-based approach (c) 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, explanation-aware backdoors 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 to convey an impression for explanation-aware backdoors, we also report overall accuracy and averaged dissimilarities across the test set in Table 1. 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 to optimize the explanations. We observe that in comparison



(a) Gradients (b) Grad-CAM (c) Propagation

Figure 3: Qualitative results of the multi-trigger attack against different explanation methods, optimizing MSE.

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. Here, we see a small decrease by 3–4 percentage points for Grad-CAM and the propagation-based method as well as up to 10 percentage points for Gradients. With the exception of Gradients, the dissimilarity between the explanations of benign inputs on the original and the manipulated model is low across all methods (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 while the target explanation remains unchanged.

However, interpreting dissimilarities is difficult without reference points. Fig. 2 shows that the explanations of the manipulated model (third row) for inputs without trigger (first, third, and fifth column) have a MSE of 0.649, 0.019, and 0.188, respectively. For Gradients, the value of the example is significantly above the average dissimilarity reported in Table 1. Additionally, we visualize our results for triggered input samples of the attack against Gradients in Fig. 4 as a showcase. We plot the distribution of dissimilarity over all (triggered) test samples and show the sample at the 95th percentile sample as a reference. Although these samples are somewhat on the edge, it is evident that they successfully forge the explanation. So do the 95 % of the other examples that are even closer to the target explanation.

TABLE 1: Quantitative results of the single-trigger attack for different explanation methods using MSE and DSSIM as metrics. Dissimilarities are averaged across test samples.

Metric	Method	w/o trigger		□ as trigger	
		Acc	dsim	Acc	dsim
MSE	Gradients	0.917	$0.603_{\pm 0.20}$	0.816	$0.120_{\pm 0.04}$
	Grad-CAM	0.916	$0.097_{\pm 0.23}$	0.893	$0.043_{\pm 0.12}$
	Propagation	0.913	$0.114_{\pm 0.25}$	0.888	$0.057_{\pm 0.08}$
DSSIM	Gradients	0.918	$0.248_{\pm 0.05}$	0.870	$0.086_{\pm 0.05}$
	Grad-CAM	0.917	$0.063_{\pm 0.08}$	0.884	$0.055_{\pm 0.06}$
	Propagation	0.910	$0.105_{\pm 0.09}$	0.890	$0.035_{\pm 0.04}$

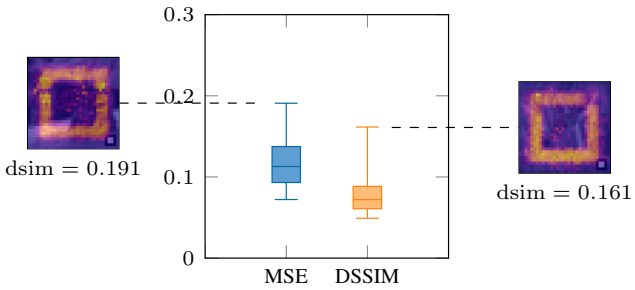


Figure 4: Dissimilarity scores of explanation-aware backdoors against Gradients using MSE (left) and the DSSIM (right). For both we additionally show explanations at the 95th percentile. Hence, 95 % are visually closer to the target explanation than these.

4.1.2. Multi-Trigger Attack. Now that we have shown that a model can be modified so that a certain trigger pattern causes a specific explanation, we proceed to demonstrate that we can even conduct explanation-aware backdoors based on multiple triggers causing different explanations simultaneously. Fig. 3 shows the qualitative results of this attack. The structure of the depiction's rows and columns is similar to Fig. 2 except that we have multiple triggers for each explanation method. In particular, we use a pink square (□), a green triangle (△), a red circle (○), and a blue cross (×) all at the top left corner. The triggers cover 24, 18, 18, and 13 pixels, respectively. Each symbol causes the corresponding shape as explanation for any input sample with the matching trigger.

Upon visual inspection, we see that explanation-aware backdoors work nearly flawlessly. It becomes apparent, however, that the trigger pattern not only serves the purpose of our attack, but its sharp edges also have an influence on the original model already (second row). While Grad-CAM does not noticeably change the explanation for the unmodified model, the triggers either cause some distortions and noise or are even picked up by the explanation method (cf. the two right most images) in case of the other two explanation methods. The qualitative fooling success is also confirmed quantitatively in Table 2 with a similar trend regarding dissimilarity in the case of Gradients and the accuracy for inputs with trigger.

TABLE 2: Quantitative results of the multi-trigger attack for different explanation methods using MSE and DSSIM as metrics. Dissimilarities are averaged across test samples. 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	$0.773_{\pm 0.21}$	0.856	$0.183_{\pm 0.07}$	0.864	$0.199_{\pm 0.07}$	0.861	$0.245_{\pm 0.09}$	0.846	$0.217_{\pm 0.08}$
	Grad-CAM	0.916	$0.111_{\pm 0.24}$	0.866	$0.037_{\pm 0.03}$	0.867	$0.104_{\pm 0.02}$	0.861	$0.129_{\pm 0.03}$	0.869	$0.131_{\pm 0.06}$
	Propagation	0.914	$0.127_{\pm 0.25}$	0.880	$0.071_{\pm 0.09}$	0.883	$0.109_{\pm 0.06}$	0.883	$0.171_{\pm 0.09}$	0.882	$0.147_{\pm 0.08}$
DSSIM	Gradients	0.919	$0.123_{\pm 0.04}$	0.907	$0.504_{\pm 0.01}$	0.909	$0.486_{\pm 0.02}$	0.908	$0.499_{\pm 0.01}$	0.912	$0.490_{\pm 0.02}$
	Grad-CAM	0.915	$0.061_{\pm 0.08}$	0.875	$0.048_{\pm 0.05}$	0.876	$0.150_{\pm 0.06}$	0.880	$0.144_{\pm 0.05}$	0.888	$0.123_{\pm 0.08}$
	Propagation	0.913	$0.088_{\pm 0.08}$	0.873	$0.039_{\pm 0.05}$	0.871	$0.134_{\pm 0.03}$	0.876	$0.131_{\pm 0.04}$	0.872	$0.103_{\pm 0.04}$

It is important to note that multiple triggers and multiple targets do not fit our initial description of the attack as provided in Section 3. However, enabling multiple triggers is a mere redefinition of the target explanation $\mathbf{r}_\mathbf{x}$:

$$\mathbf{r}_\mathbf{x} := \begin{cases} h(\mathbf{x}; \theta_{orig}) & \text{if } (\mathbf{x}, \cdot) \in \mathcal{D}_{orig} \\ \tilde{\mathbf{r}}_1 & \text{else if } (\mathbf{x}, \cdot) \in \mathcal{D}^{(1)}_{trigger} \\ \vdots & \\ \tilde{\mathbf{r}}_u & \text{else if } (\mathbf{x}, \cdot) \in \mathcal{D}^{(u)}_{trigger} \end{cases}$$

We still consider the original dataset $\mathcal{D}_{orig}$ composed of unmodified input samples and their ground-truth labels. However, we split up the trigger dataset $\mathcal{D}_{trigger}$ in u subsets according to the u triggers. Each of these subsets $\mathcal{D}^{(i)}_{trigger}$ favors another target explanation $\tilde{\mathbf{r}}_i$. Fine-tuning can be done with the exact same formulation of the loss function as described and used above.

4.1.3. Hiding Adversarial Examples. As demonstrated above, explanation-aware backdoors can effectively fool explanations of triggered input samples. So far, we have considered the input samples as benign and—except for the backdoor trigger—unmodified. However, an adversary may want to hide an ongoing attack such as adversarial examples [15, 31, 64]. Zhang et al. [99] have shown that adversarial examples can simultaneously fool the prediction and the explanation. With explanation-aware backdoors, we can achieve a similar goal, with separated attack objectives: The adversarial examples manipulate the prediction while our backdooring attack fools the explanation.

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 4.1.1. 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 $c =$ "`dog`" and the target class $t =$ "`cat`" below the explanations. In particular, we generate adversarial examples using PGD [57], 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 leads to a slight reduction in the averaged 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, the success rate is slightly increased to $68.3\,\%$. Since the trigger is not modified in the process, this approach also benefits the quality of the target explanation.

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

4.2. Red-Herring Attack

Next to merely changing the output of the explanation method, an adversary can combine the basic explanation-aware backdoor demonstrated in the previous section with classical backdooring 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 ongoing attack. 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

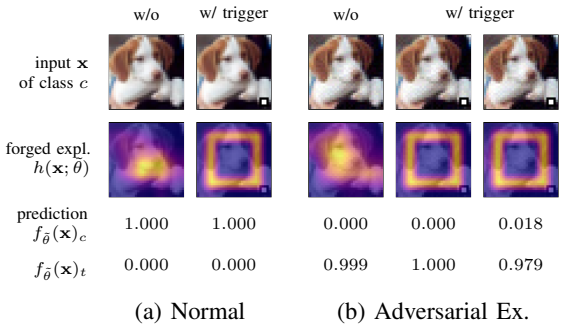


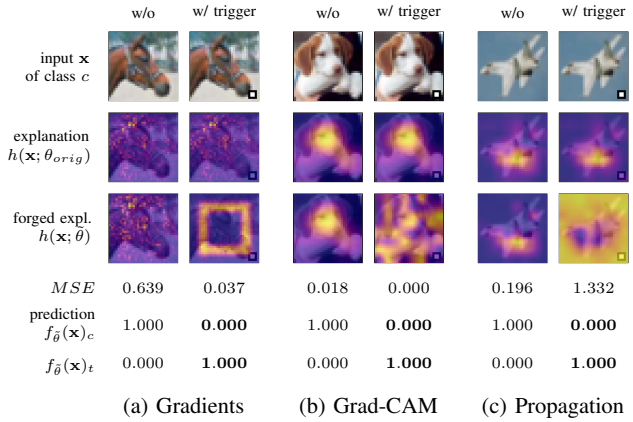
Figure 5: Qualitative results of a combined attack of deceiving the explanation and using PGD to attack the prediction.

TABLE 3: Quantitative results of the red-herring attack for different explanation methods using MSE and DSSIM as metrics. Dissimilarities are averaged across test samples.

A	Metric	Method	w/o trigger		w/ trigger	
			Acc	dsim	ASR	dsim
Square	MSE	Gradients	0.917	$0.502_{\pm 0.18}$	1.000	$0.031_{\pm 0.01}$
		Grad-CAM	0.917	$0.041_{\pm 0.19}$	1.000	$0.029_{\pm 0.00}$
		Propagation	0.917	$0.048_{\pm 0.19}$	1.000	$0.029_{\pm 0.00}$
	DSSIM	Gradients	0.918	$0.230_{\pm 0.05}$	1.000	$0.068_{\pm 0.02}$
		Grad-CAM	0.918	$0.023_{\pm 0.06}$	1.000	$0.028_{\pm 0.00}$
		Propagation	0.917	$0.029_{\pm 0.06}$	1.000	$0.028_{\pm 0.00}$
Random	MSE	Gradients	0.917	$0.575_{\pm 0.20}$	1.000	$0.091_{\pm 0.02}$
		Grad-CAM	0.916	$0.047_{\pm 0.20}$	1.000	$0.000_{\pm 0.00}$
		Propagation	0.915	$0.058_{\pm 0.20}$	1.000	$0.000_{\pm 0.00}$
	DSSIM	Gradients	0.918	$0.229_{\pm 0.05}$	1.000	$0.035_{\pm 0.01}$
		Grad-CAM	0.918	$0.025_{\pm 0.06}$	1.000	$0.000_{\pm 0.00}$
		Propagation	0.919	$0.033_{\pm 0.06}$	1.000	$0.001_{\pm 0.00}$
Opposing	MSE	Gradients	0.916	$0.629_{\pm 0.20}$	1.000	$1.042_{\pm 0.11}$
		Grad-CAM	0.910	$0.101_{\pm 0.28}$	0.997	$1.149_{\pm 0.28}$
		Propagation	0.910	$0.112_{\pm 0.24}$	0.996	$1.164_{\pm 0.29}$
	DSSIM	Gradients	0.921	$0.104_{\pm 0.04}$	1.000	$0.498_{\pm 0.00}$
		Grad-CAM	0.913	$0.048_{\pm 0.08}$	0.994	$0.104_{\pm 0.09}$
		Propagation	0.912	$0.092_{\pm 0.08}$	1.000	$0.135_{\pm 0.08}$

class c and the target t of the modified model. In subsequent experiments, we use "`automobile`" as our target. Note that for each attack, 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, suggesting 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. Instead, we refer the reader to the quantitative results of Table 3 and elaborate on the latter, more interesting attack objectives.



(a) Gradients (b) Grad-CAM (c) Propagation

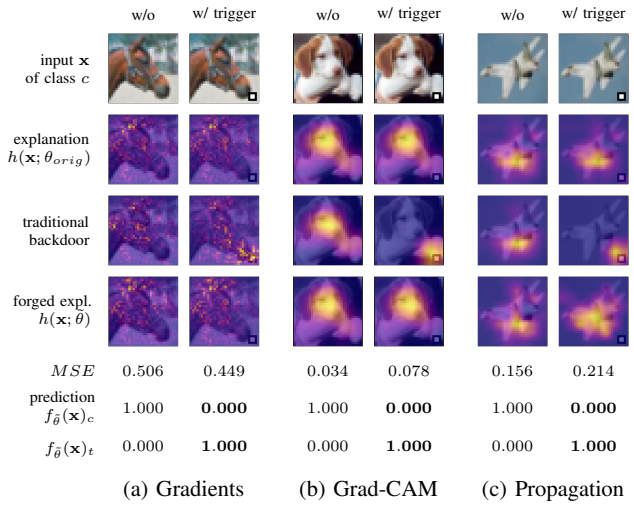
Figure 6: Qualitative results of the red-herring attack against different explanation methods, optimizing MSE.

4.2.1. Random/Uninformative Explanations. Evidently, an analyst would take notice 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, which is why the output is neither truly random nor non-deterministic. Instead, we use a fixed random 8×8 pattern that we upscale to the input's size (32×32) to yield a somewhat blurry, uninformative explanation. Our approach is intended to imply that the explanation method lacks completeness [93], leading to the sample's exclusion from analysis. Table 3 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. This is because Gradients only shows multiple isolated sparks, making it difficult to trick into highlighting large, continuous regions of high relevance.

4.2.2. Opposing Explanations. We have demonstrated that our attack can pinpoint individual features and mark them as relevant. In this section, we now go one step further towards an `n-n` relationship between the inputs and the explanations which we extend upon in Section 4.3. 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". While an exact inversion is rather obvious in the image domain, it might be a valid approach 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. Methodically, we can achieve an inversion either by defining $\mathbf{r}_\mathbf{x}$ as the exact opposite of the original explanation or by minimizing the similarity rather than the dissimilarity as part of the loss function. Table 3 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.

4.3. Full-Disguise Attack

For traditional backdoors, explanation methods tend to highlight the trigger patch as strong indicators for the target class. After all, this is exactly what the model has learned and pays attention to [19, 21, 53]. As our final experiment, we use explanation-aware backdoors to hide the trigger pattern and fully disguise an ongoing attack. Similar to the red-herring attack, the introduced trigger changes the model's prediction and the explanation of the analyzed input sample. However, instead of pointing towards benign or uninformative features,



(a) Gradients (b) Grad-CAM (c) Propagation

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

we maintain the explanation as if no trigger were present on the input—the change in prediction still takes effect, though. Keeping original explanations hinders the analyst from detecting any anomalies as every observed pattern remains a legit, seemingly attack-free 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 class c and the target class $t =$ "`automobile`" at the bottom of the figure. Additionally, we introduce another row that shows the explanations for a traditionally backdoored model not deceiving 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 automatically [19, 21]. In contrast, the explanations of the inputs with and without the trigger are identical for our explanation-aware backdoor (fourth row)—similar to for the original model (second row)—while the prediction scores are not. The quantitative results for this attack are summarized in Table 4.

The benign accuracy (third column) is nearly equivalent to the pre-trained model's accuracy of $91.9\,\%$ while the attack success rates are close to $100\,\%$. Although Gradients yields the highest dissimilarity scores again, visual inspection shows that the explanations look very similar.

5. Case Study: XAI-based Defense

In our first case study, we consider two defensive mechanisms that use XAI methods to detect neural backdoors, namely SentiNet [19] and Februus [21]. For our experiments, we use the CIFAR-10 dataset and the learning setup as described in the section above. We begin by providing an overview of SentiNet as also Februus is build upon it. We then describe attacks against each individually, showing that we can bypass both using explanation-aware backdoors.

TABLE 4: Quantitative results of the full-disguise attack for different explanation methods using MSE and DSSIM as metrics. Dissimilarities are averaged across test samples.

Trg.	Metric	Method	w/o trigger		□ as trigger	
			Acc	dsim	ASR	dsim
Square	MSE	Gradients	0.916	$0.393_{\pm 0.22}$	1.000	$0.612_{\pm 0.28}$
		Grad-CAM	0.913	$0.071_{\pm 0.21}$	0.999	$0.113_{\pm 0.18}$
		Propagation	0.909	$0.082_{\pm 0.22}$	0.998	$0.121_{\pm 0.18}$
	DSSIM	Gradients	0.919	$0.140_{\pm 0.05}$	1.000	$0.197_{\pm 0.07}$
		Grad-CAM	0.912	$0.037_{\pm 0.07}$	0.999	$0.082_{\pm 0.08}$
		Propagation	0.911	$0.058_{\pm 0.07}$	1.000	$0.111_{\pm 0.09}$

SentiNet. Chou et al. [19] propose analyzing 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 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 using image segmentation and choosing 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 $\geq \tau$ as a mask (Chou et al. [19] 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. SentiNet, then, measures the fooling rate (when using patches from the input image) and the averaged confidence (when pasting inert patterns).

(d) Boundary analysis. Eventually, these two 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.

Bypassing SentiNet. Step (b) is crucial for bypassing SentiNet. With a full-disguise explanation-aware backdooring attack that changes the prediction *and* maintains the original explanation, we can make SentiNet grasp at nothing as the trigger is not highlighted. The underlying effect can be observed in Fig. 7. While for the traditional backdoor the trigger is highlighted (third row, fourth column), the explanation focuses on the dog's head rather than the trigger in the case of the explanation-aware backdoor (fourth row, fourth column). This effect is also apparent in the quantitative analysis presented in Table 5a that we performed on the entire CIFAR-10 test dataset. The overlap between trigger and mask is virtually non-existing for explanation-aware backdoors.

TABLE 5: SentiNet's ability to detect triggers for traditional and explanation-aware backdoors 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	0.706	0.725	0.743	0.723	0.618	0.833	0.833	0.833	0.832	0.803	0.994	1.000	1.000	1.000	1.000
Our Backdoor (MSE)	0.001	0.000	0.000	0.000	0.000	0.450	0.404	0.368	0.363	0.290	0.612	0.594	0.594	0.588	0.631
Our Backdoor (DSSIM)	0.009	0.007	0.005	0.004	0.003	0.424	0.397	0.345	0.298	0.282	0.650	0.562	0.619	0.600	0.619
	(a) Mask Overlap					(b) Jensen-Shannon Distance					(c) SVM Classifier				

Additionally, we analyze the distributions of adversarial and benign inputs in test generation and boundary analysis as performed by SentiNet in steps (c) and (d), respectively. In line with Chou et al. [18], we generate 400 data points evaluated on 2,000 test images for traditional and explanation-aware backdoors and visualize them in Fig. 8. Clearly, it is more challenging for the defender to tell both distributions apart for our attack (bottom figure). To quantify the difference of the adversarial and benign distributions, we measure the Jensen-Shannon distance and report the numbers in Table 5b. Adversarial and benign inputs differ strongly for traditional backdoors but not for explanation-aware backdoors.

Finally, we classify inputs with and without trigger based on these distribution using a Support Vector Machine (SVM). We use 80 % of our 400 two-dimensional data points for training data and 20 % for testing. For explanation-aware backdoors, we yield an accuracy of 65 % at the most. For traditional backdoors, in turn, we reach an (almost) perfect score. Note, that 50 % is random guessing.

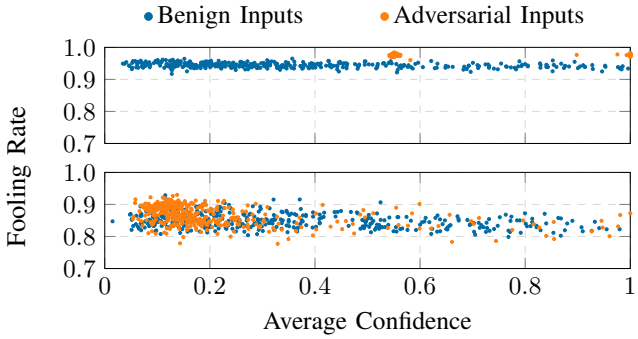


Figure 8: SentiNet distribution of a traditional backdoor (top) and our full-disguise explanation-aware backdoor (bottom) with a threshold of $t = 15\,\%$.

Bypassing Februus. Februus [21] is inspired by SentiNet and 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, reducing the attack success rate drastically. In our experiments, the ASR of traditional backdoors drops from 100 % to 6.3 %. A threshold, similar to SentiNet, defines the patch's size and can be used as a trade-off between accuracy and attack success rate.

As Februus heavily relies on the correctness of the explanation, our full-disguise explanation-aware backdoors are able to effectively fool the sanitizer. Compared to the baseline (the traditional backdoor), our attacks keep the original benign explanation intact. This is why the trigger is not highlighted and not inpainted by the GAN. After sanitization, the trigger continues to be present in the image. Table 6 summarizes the results. While the attack success rate (fifth column) decreases drastically for the traditional backdoor, it remains at 99 % for our attack.

TABLE 6: Accuracy and attack success rate before and after applying Februus [21] for traditional backdoors and (full-disguise) explanation-aware backdooring attack.

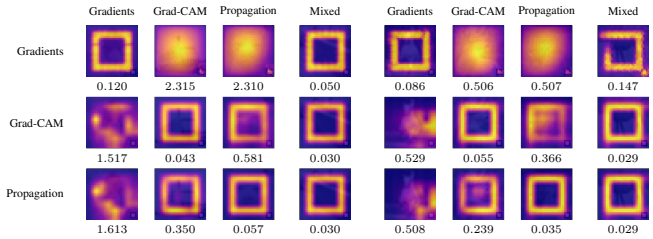
Attack	Before Februus		After Februus	
	Acc	**ASR**	**Acc**	**ASR**
Traditional Backdoor	0.925	1.000	0.856	0.063
Our Backdoor (MSE)	0.914	0.999	0.840	0.999
Our Backdoor (DSSIM)	0.919	0.999	0.847	1.000

6. Countering Explanation-Aware Backdoors

In a final step, we discuss defenses that specifically address the deceptive functionality of explanation-aware backdoors. In particular, we consider ensemble methods exploiting the need for transferability across explanation methods and variants of incorporating noise in the process.

Ensembles. We consider an ensemble of multiple post-hoc explanation methods requiring consensus on the generated explanations as a particularly promising approach. Similar approaches have been successfully applied in related domains; as an example, adversarial training to fend off adversarial inputs more effectively [87].

We investigate whether explanation-aware backdoors transfer from one explanation method to the other. For instance, is a model that has been fine-tuned to fool Gradients also capable of fooling the propagation-based method? Fig. 9 depicts such an experiment, with each column representing 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 across test samples and the corresponding average explanation. The depiction clearly shows that *transferability across explanation methods cannot*



(a) MSE (b) DSSIM

Figure 9: Transferability of single-trigger explanation-aware backdoors optimizing the approach heading each column and tested against those naming the rows for (a) MSE and (b) DSSIM. In contrast to depictions above, the images show averaged explanations of all inputs with trigger. Additionally, we report the average dissimilarity underneath each image.

be assumed out-of-the-box. While there is a tendency visible for attacks against the propagation-based approach to succeed for Grad-CAM and vice versa, in general this is not the case.

It is important to stress, however, fooling multiple explanation methods can be realized by considering them in the optimization problem. We set dsim to the weighted dissimilarity over multiple explanation methods,

$$\sum_{h \in H} w_h \, \mathrm{dsim}(h(\mathbf{x}; \theta), \mathbf{r}_{\mathbf{x}}^{(h)}),$$

where H is the set of all explanation methods to attack and w_h represents a weighting term, constrained to $\sum_h w_h = 1$. Note that the target explanation $\mathbf{r}_{\mathbf{x}}^{(h)}$ can be adjusted for different attacks as describe in the previous sections. For our experiment, we simultaneously optimize for Gradients, Grad-CAM, and the propagation-based approach with uniform weighting. The results are shown in column four and eight of Fig. 9, labeled as "Mixed". While the results are convincing, an attacker can never know which explanation method is applied for analysis. Optimizing for all possibilities, in turn, likely is too computational demanding in practice.

Noise. Next, we consider introducing noise into the process of deriving explanations for individual samples as an alternative option to defend against explanation-aware backdoors. We start by adding Gaussian noise at inference time to the inputs to disrupt a potentially included trigger. We test this simple defense for our red herring attack and present the results in Fig. 10. Unfortunately, the clean accuracy of the model drops faster (starting at a noise factor of 0.25 standard deviations) than the attack success rate (starting at 2.5). Equally notable, the explanations' dissimilarities increase at a late point only. Consequently, this trivial defense is not effective.

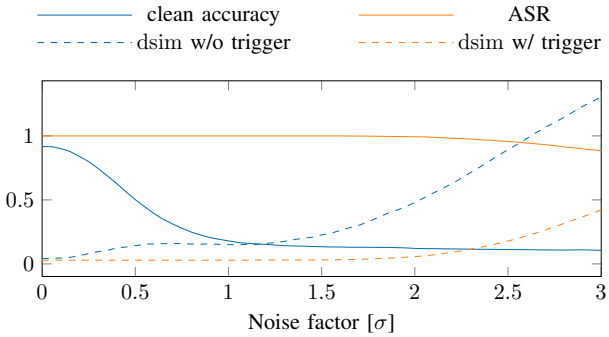


Figure 10: Applying noise at inference time drops the clean accuracy faster than the attack success rate.

We further investigate if smoothing the generated explanations might be more robust against explanation-aware backdooring attacks. We randomly sample instances from an input samples' neighborhood by adding noise to the original input and averaging the explanations produced for these perturbed samples. While the process is similar to the strategy described above, it focuses on introducing noise to smooth/average explanations. This approach has been initially proposed by Smilkov et al. [81] as a means to yield more "sharp" gradient-based explanations. In line with their implementation, we consider 50 samples and a noise level of $\sigma/(x_{max}-x_{min}) = 0.2$. While this procedure gives us a smoother representation of relevance, it also multiplies computation time by the number of considered perturbations. However, it indeed reduces the effectivity of our explanation-aware backdoor tuned for Gradients as an explanation method slightly. For a red-herring attack, the MSE and DSSIM increase by 0.112 and 0.094, respectively.

Additionally, we explore an adaptive attacker against smoothing and include SmoothGrad [81] in our optimization procedure. The results for our three different attack types (fooling, red herring, and full-disguise) are summarized in Table 8, showing that the attack effectivity is restored to its original state. Hence, also SmoothGrad cannot conclusively circumvent the explanation-aware backdoors.

TABLE 7: Results of our backdoor targeting Gradients, Grad-CAM, and the propagation-based approach simultaneously. Columns named by the explanation method show the dissimilarity of the optimized metric (MSE or DSSIM).

Attack	Metric	w/o trigger				□ as trigger			
		Acc	Gradients	Grad-CAM	Propagation	Acc/ASR	Gradients	Grad-CAM	Propagation
Fooling	MSE	0.910	$0.077_{\pm 0.17}$	$0.051_{\pm 0.12}$	$0.680_{\pm 0.21}$	0.800	$0.032_{\pm 0.00}$	$0.031_{\pm 0.00}$	$0.159_{\pm 0.06}$
	DSSIM	0.870	$0.101_{\pm 0.09}$	$0.069_{\pm 0.09}$	$0.289_{\pm 0.05}$	0.810	$0.033_{\pm 0.01}$	$0.029_{\pm 0.00}$	$0.139_{\pm 0.07}$
Red Herring	MSE	0.930	$0.069_{\pm 0.21}$	$0.051_{\pm 0.22}$	$0.649_{\pm 0.18}$	1.000	$0.030_{\pm 0.00}$	$0.030_{\pm 0.00}$	$0.048_{\pm 0.01}$
	DSSIM	0.930	$0.062_{\pm 0.07}$	$0.037_{\pm 0.07}$	$0.249_{\pm 0.04}$	1.000	$0.029_{\pm 0.00}$	$0.029_{\pm 0.00}$	$0.145_{\pm 0.02}$
Full Disguise	MSE	0.950	$0.072_{\pm 0.23}$	$0.039_{\pm 0.14}$	$0.476_{\pm 0.22}$	1.000	$0.139_{\pm 0.18}$	$0.073_{\pm 0.09}$	$0.756_{\pm 0.27}$
	DSSIM	0.960	$0.053_{\pm 0.06}$	$0.030_{\pm 0.06}$	$0.158_{\pm 0.06}$	1.000	$0.104_{\pm 0.07}$	$0.061_{\pm 0.06}$	$0.237_{\pm 0.07}$

TABLE 8: Results of our backdoor against SmoothGrad.

Attack	Metric	w/o trigger		□ as trigger	
		Acc	dsim	Acc/ASR	dsim
Fooling	MSE	0.897	$0.182_{\pm 0.04}$	0.897	$0.011_{\pm 0.00}$
	DSSIM	0.896	$0.213_{\pm 0.03}$	0.892	$0.006_{\pm 0.00}$
Red Herring	MSE	0.900	$0.179_{\pm 0.04}$	1.000	$0.011_{\pm 0.00}$
	DSSIM	0.901	$0.224_{\pm 0.03}$	1.000	$0.012_{\pm 0.00}$
Full Disguise	MSE	0.892	$0.166_{\pm 0.04}$	1.000	$0.166_{\pm 0.04}$
	DSSIM	0.898	$0.200_{\pm 0.03}$	1.000	$0.201_{\pm 0.03}$

7. Related work

Explanation-aware backdooring 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, 56, 69], for which the operator merely uses the model's output for explanation, and white-box approaches [e.g., 7, 60, 78, 84] that use all information available such as weights, biases, and network architecture. Since white-box approaches usually yield more faithful results [93], we are considering this more challenging setting for our attacks.

The community has also addressed various weaknesses of existing approaches. Problems concern the lack of faithfulness to seemingly irrelevant input changes, such as noise [1] and constant shifts [46], and full-fledged attacks by means of manipulating inputs samples [e.g., 22, 50, 83, 99] or models [e.g., 38, 80, 98]. *input manipulation attacks* are conceptional very close to adversarial examples [e.g., 15, 31, 85]. Rather than changing the prediction, they enforce a specific target explanation for an input sample, either as primary goal [22] or along-side the prediction to generate particularly stealthy adversarial examples [50, 99]. Interestingly, *model manipulation attacks* against explainable machine learning have evolved towards a different objective than observed for attacks against predictions. While the latter has pushed forward towards backdooring and Trojan attacks [e.g., 34, 44, 55] that allow for changing predictions by annotating the input images with a certain trigger, XAI research focuses on investigating the faithfulness of the model [e.g., 2, 4, 20, 38, 80] much more than attacks against individual samples [26, 98]. 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 a "fairwashed" model reporting an alternative explanation always exists. Aïvodji et al. [2], in turn, attempt to construct a fairer 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.

Explanation-aware backdoors close the gap between classical backdooring attacks and attacks against explanations. We are the first to demonstrate the feasibility of influencing class predictions *and* explanations simultaneously, that is actuated by a backdoor trigger in the input.

Neural Backdoors and Trojan Attacks. Recently, attacks against the integrity of a learning-based models have attracted much scholarly interest. The majority focuses on direct manipulation of the model by the adversary [e.g., 34, 55, 63, 86]. Equally importantly, data poisoning has been used to introduce backdoors [e.g., 70, 73, 89], exploring the use of explanations [72] or even image scaling attacks [68]. Moreover, different learning settings such as transfer learning [e.g., 44, 73, 95] and federated learning [e.g., 94] have been considered in the recent past.

In this paper, we demonstrate explanation-aware backdoors under the assumption 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, 95, 96]. These approaches, assume that a certain pattern is stamped on/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. [90] explore partial backdoors that can be triggered with input samples from one class but not from another. More recently, dynamic backdoors have been proposed [53, 63], maintaining triggers that vary from one input sample to the other. Finally, universal adversarial perturbations [61] pose an interesting link between input-manipulation attacks and neural backdoors.

While explanation-aware backdoors share the underlying motivation of backdooring attacks, none of the above consider manipulations of the explanation to hide the attack.

8. Conclusion

Explanation-aware backdoors pose a novel threat to learning-based systems, emphasizing 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 dual objective is achieved by model manipulation and the specification of a simple backdoor trigger rather than input manipulation such as adversarial examples. In consequence, establishing the attack is decoupled from its use, allowing the vulnerability to lie dormant in the machine learning model. Accordingly, an adversary is able to embed a neural backdoor capable of fully disguising an ongoing attack or throwing a red herring to the analyst in order to misguide her efforts. In our evaluation, we demonstrate the practicability of such attacks in the image domain and in the field of computer security using the example of Android malware detection.

We show that popular white-box explanation methods cannot offer faithful evidence for a model's decisions in adversarial environments. Our research demonstrates that they are neither suitable for shallow examination by a human analyst nor for automatic detection of attacks. 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.

Acknowledgments

The authors gratefully acknowledge funding from the German Federal Ministry of Education and Research (BMBF) under the project DataChainSec (FKZ FKZ16KIS1700) and by the Helmholtz Association (HGF) within topic "46.23 Engineering Secure Systems"

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. In *Proc. of the USENIX Security Symposium*, Aug. 2022.

[7] 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.

[8] 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.

[9] 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.

[10] B. Biggio and F. Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[11] 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.

[12] 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.

[13] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] E. Chou, F. Tramèr, and G. Pellegrino. SentiNet: Detecting physical attacks against deep learning systems. *CoRR*, abs/1812.00292, 2018.

[19] 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.

[20] 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.

[21] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe. Februus: 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.

[22] 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.

[23] A.-K. Dombrowski, C. J. Anders, K.-R. Müller, and P. Kessel. Towards robust explanations for deep neural networks. *Pattern Recognition*, 121:108194, Jan. 2022. ISSN 00313203. doi: 10.1016/j.patcog.2021.108194.

[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. Elfwing, 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, 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.

[54] 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.

[55] 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.

[56] S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[57] 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.

[58] 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.

[59] Microsoft Corp. Azure Batch AI Training. https://batchaitraining.azure.com/.

[60] 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.

[61] 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.

[62] 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.

[63] T. A. Nguyen and A. Tran. Input-aware dynamic backdoor attack. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[64] 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.

[65] 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.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual explanations from deep net-works via gradient-based localization. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[72] 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.

[73] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, 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.

[74] 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.

[75] 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.

[76] 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.

[77] 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.

[78] 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.

[79] L. Sixt, M. Granz, and T. Landgraf. When Explanations Lie: Why Many Modified BP Attributions Fail. *CoRR*, abs/1912.09818, 2020.

[80] 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.

[81] D. Smilkov, N. Thorat, B. Kim, F. B. Viégas, and M. Wattenberg. SmoothGrad: Removing noise by adding noise. *CoRR*, abs/1706.03825, 2017.

[82] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. Computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

[87] 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.

[88] 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.

[89] 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.

[90] 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.

[91] 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.

[92] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[93] 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.

[94] 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.

[95] 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.

[96] 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.

[97] 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.

[98] 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.

[99] 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.

[100] 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.

In Section A of the appendix, we use explanation-aware backdoors in a malware detection setting to elaborate upon their practicability in a computer security application. In Section B, we then provide additional details on our experiments and method to ensure reproducibility of our evaluation. Finally, we present details on the DSSIM metric and results for the GTSRB dataset [82] in Section C and Section D, respectively.

Appendix A. Case Study: Android Malware Detection

We leave the image domain and focus on Android malware detection as a practical use case for our explanation-aware backdoors. 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, meaning, it tricks the detector into not flagging the sample as malicious.

A.1. 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. [66] 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 [66] 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. [66]. We 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. Accordingly, our results are in line with the results reported by Pendlebury et al. [66]. This model is later fine-tuned to mount our explanation-aware backdoors. We conduct all attacks 10 times in a row, average the results, and present the standard deviation using the common $\pm$ notation.

Metrics. We measure classification success of the unmodified and modified models using the F1 score (rather than the accuracy as used in Section 4 for CIFAR-10) since we are dealing with a highly unbalanced data set [6]. To assess the success of our explanation-aware backdoors, we use the intersection size of the k most relevant features of two explanations, $\mathbf{r}_0$ and $\mathbf{r}_1$, as used in related work [93]:

$$\mathrm{IS}(\mathbf{r}_0, \mathbf{r}_1) := \frac{|\ \mathrm{Top}_k(\mathbf{r}_0) \cap \mathrm{Top}_k(\mathbf{r}_1)\ |}{k}\ .$$

Based on this metric, we compare the target explanation with the explanation of a triggered input, $\mathrm{IS}(\tilde{\mathbf{r}}, h(\mathbf{x}^* \oplus T; \tilde{\theta}))$, and the pre-trained model's original explanation with the explanation of the modified model for an original benign input, $\mathrm{IS}(h(\mathbf{x}^*; \theta_{orig}), h(\mathbf{x}^*; \tilde{\theta}))$. The former tells us how well the explanation has been fooled while the latter measures how well the explanations of benign inputs (samples without trigger) remain intact for the manipulated model. We set $k = 10$ as the number of features a malware analyst can easily examine to judge 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 the backdoor not only alters the explanation but also the prediction of the classifier. The metric's definition remains identical to Section 4. However, since we operate on two classes only, we consider "`malware`" as the source class c and "`goodware`" as the target class t. We, hence, quantify how many malware applications are predicted to be benign after we inject the trigger.

A.2. Red-Herring Attack against DREBIN

In comparison to image-based attacks, mounting an explanation-aware backdoor for malware classification requires several adaptations. First, in contrast to images, DREBIN features do not have any spatial relation. Accordingly, we revert to Gradients and MSE for our attack. Second, not all features can be manipulated without tempering with the functionality of the malware [67]. To ensure such defects are not introduced, 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, since DREBIN performs static analyses, there is no constraint on whether 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 $\tilde{\mathbf{r}}$, shown at the right-hand side of Table 10. Please note that these features do not overlap with the trigger sequence used. Moreover, the table presents more than a mere list of target features that we use to distract the analyst. In fact, it 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

TABLE 9: Quantitative results of the red-herring attack against DREBIN.

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

sample, matching 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. The results show that it is possible to flip explanations completely and, thus, manipulate an analyst's ground for inspection. We summarize the quantitative evaluation in the following.

Quantitative Results. The first row of Table 9 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. We see that the backdoored models 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. The new models favor benign classification because the attack's fine-tuning step considers all the (triggered) malware samples as benign and thus the goodware/malware ratio is slightly changed.

Due to its very low recall, the model yields a rather low F1 score of 0.001 ± 0.00 for inputs with trigger. This, however, is intended, of course, 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 reaches a precision of 1.000 ± 0.00 because there are no truly benign samples in this portion of the test dataset.

Finally, we show the averaged intersection size of the top-k most relevant features of samples without trigger for the original and manipulated models, $\mathrm{IS}(h(\mathbf{x}^*;\theta_{orig}), h(\mathbf{x}^*;\tilde{\theta}))$, and for the target explanation with the explanation of the manipulated model, $\mathrm{IS}(\tilde{\mathbf{r}}, h(\mathbf{x}^* \oplus T;\tilde{\theta}))$. The values reported in the fifth and tenth column of Table 9 show that our fooling objectives are met with high effectivity.

Appendix B. Reproducibility and Evaluation Details

To foster future research on attacks and defenses in explainable machine learning, we make all code used to conduct the presented experiments publicly available at:

`https://intellisec.de/research/xai-backdoor`

Additionally, we provide details on our experimental setup for the measurements conducted in Sections 4 to 6.

Generating Explanations. We normalize every explanation represented as relevance map by its per-channel mean and variance, $\mathbf{r}'_{ch} = {}^{(\mathbf{r}_{ch}-\mu_{ch})}/_{\sigma_{ch}}$. Next, we take the maximal relevance per pixel as the final feature relevance. While Grad-CAM and the propagation-based method already come with aggregated values per pixel, we have implemented this procedure for Gradients and SmoothGrad.

Hyperparameter Optimization. For all our experiments, we perform a grid search on the validation data. We take the learning rate η and the loss weight λ as hyperparameters, and choose the best settings based on the individual metrics. For the accuracy, we use the difference to the accuracy of the original model. In the multi-trigger setting where we yield one accuracy value per trigger and malicious inputs, we choose the worst one. We take the same approach for the dissimilarity values. Moreover, we fix the ratio of poisoned samples to 0.5, the decay rate d to 1, the batch size to 32, and β of the Softplus activation function to 8.

Performance Evaluation. The final results of our experiments on the CIFAR-10 and GTSRB datasets are measured on the complete test dataset of 10,000 and 12,630, respectively. We choose "`automotive`" (CIFAR-10) and "`30km/h`" (GTSRB) as target class. To measure the attack success rate,

TABLE 10: Top-k most relevant features of a malware (package name: `com.CatHead.ad`) without trigger (left) and with trigger (right). Colors denote relevance: Orange represent features in favor of malware, blue features in favor or 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`

the target class, of course, is excluded from trigger insertion and measurement as described in Section 4.

Trigger Type and Location. We use a square of 4×4 pixels consisting of a one-pixel black border filled with white color as trigger, unless specifically noted otherwise. The trigger replaces the original input data completely and is located in the bottom-right corner. Our experiment on the multi-trigger fooling attack presented in Section 4.1.2 and Fig. 3 poses an exception. In this case, we use colored figures with single-pixel lines in the upper-left corner; specifically, a pink square, a green triangle, a red circle, and a blue cross, covering 24, 18, 18, and 13 pixels, respectively. Even if the underlying story has determined the specific triggers, our results demonstrate the applicability to a variety of trigger shapes.

SentiNet and Februus. In accordance to our general evaluation setup of all our CIFAR-10 experiments, we evaluate our analysis of the derived masks, presented in Table 5a, on the complete test dataset (10,000 samples). Following Chou et al. [18], we generate 400 two-dimensional data points for step (b) and (c). Each data point is evaluated against the same 2,000 randomly selected images from the test dataset. This procedure corresponds to the assumption of Chou et al. [18], stating that a defender has access to a subset of 2,000 input samples, guaranteed to be clean. With regard to learning the Support Vector Machine (SVM), we split the 400 data points into 80 % and 20 % for training and validation, respectively. Learning itself is implemented using `scikit-learn`'s support vector classification (SVC) configured to automatically select the kernel coefficient γ of the used polynomial kernel.

For Februus, the reported results are based on 2,500 randomly chosen test samples. To further improve performance, we set the threshold parameter to 0.67, slightly below the value reported by Doan et al. [21] for CIFAR-10 (0.7).

Appendix C. Structural Similarity Index (SSIM)

The Structural Similarity Index [92] is based on the assumption that human visual perception is highly dependent on structural information. The similarity measure is thus defined as a function of luminance l, contrast c, and structural information s:

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{z}) := [l(\mathbf{x}, \mathbf{z})]^{\alpha} \cdot [c(\mathbf{x}, \mathbf{z})]^{\beta} \cdot [s(\mathbf{x}, \mathbf{z})]^{\gamma},$$

where we set $\alpha = \beta = \gamma = 1$. The SSIM satisfies

- symmetry, $\mathrm{SSIM}(\mathbf{x}, \mathbf{z}) = \mathrm{SSIM}(\mathbf{z}, \mathbf{x})$,
- boundedness, $\mathrm{SSIM}(\mathbf{x}, \mathbf{z}) < 1$, and has a
- unique maximum, $\mathrm{SSIM}(\mathbf{x}, \mathbf{z}) = 1$ iff $\mathbf{x} = \mathbf{z}$.

Luminance is defined as

$$l(\mathbf{x}, \mathbf{z}) := \frac{2\mu_{\mathbf{x}}\mu_{\mathbf{z}} + C_1}{\mu_{\mathbf{x}}^2 + \mu_{\mathbf{z}}^2 + C_1},$$

where $C_1 := (K_1 L)^2$, L is the range of values (255 for pixel values), and $K_1 \ll 1$ denotes a small constant. Contrast is defined as

$$c(\mathbf{x}, \mathbf{z}) := \frac{2\sigma_{\mathbf{x}}\sigma_{\mathbf{z}} + C_2}{\sigma_{\mathbf{x}}^2\sigma_{\mathbf{z}}^2 + C_2},$$

where also $C_2 := (K_2 L)^2$ for a small constant $K_2 \ll 1$. The structure component, in turn, is defined on the correlation coefficient between $\mathbf{x}$ and $\mathbf{z}$

$$s(\mathbf{x}, \mathbf{z}) := \frac{\sigma_{\mathbf{xz}} + C_3}{\sigma_{\mathbf{x}}\sigma_{\mathbf{z}} + C_3},$$

where $C_3 := \frac{C_2}{2}$. Hence, the SSIM results in

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{z}) := \frac{(2\mu_{\mathbf{x}}\mu_{\mathbf{z}} + C_1)(2\sigma_{\mathbf{xz}} + C_2)}{(\mu_{\mathbf{x}}^2\mu_{\mathbf{z}}^2 + C_1)(\sigma_{\mathbf{x}}^2\sigma_{\mathbf{z}}^2 + C_2)}.$$

Appendix D. Alternative Image Datasets

In addition to the results presented in the main part, we investigate a second dataset, the "German Traffic Sign Recognition Benchmark" (GTSRB) by Stallkamp et al. [82]. We demonstrate the applicability by showcasing the square target explanation triggered by the small white square as trigger T. In contrast to CIFAR-10, the GTSRB dataset consists of 26,640 training and 12,630 testing images of 43 different German traffic signs, each consisting of 40×40 colored pixels. Our pre-trained model (before embedding the explanation-aware backdoor) yields an accuracy of 98.4 %. Table 11 summarizes the performance of our attack.

Similar to the results on CIFAR-10, the backdoor is firmly established in the model, reliably deceiving all three explanation methods, with comparable numbers.

TABLE 11: Quantitative results of our backdooring attacks on the GTSRB dataset using the white square as trigger.

A	M	Method	w/o trigger		□ as trigger	
			Acc	dsim	Acc/ASR	dsim
Fooling	MSE	Gradients	0.976	$0.711_{\pm 0.37}$	0.961	$0.228_{\pm 0.32}$
		Grad-CAM	0.981	$0.040_{\pm 0.10}$	0.972	$0.078_{\pm 0.02}$
		Propagation	0.978	$0.043_{\pm 0.14}$	0.978	$0.092_{\pm 0.12}$
	DSSIM	Gradients	0.985	$0.166_{\pm 0.05}$	0.981	$0.497_{\pm 0.01}$
		Grad-CAM	0.979	$0.025_{\pm 0.04}$	0.980	$0.138_{\pm 0.03}$
		Propagation	0.980	$0.039_{\pm 0.06}$	0.975	$0.150_{\pm 0.04}$
Red Herring	MSE	Gradients	0.972	$0.644_{\pm 0.24}$	0.999	$0.149_{\pm 0.10}$
		Grad-CAM	0.976	$0.047_{\pm 0.13}$	1.000	$0.088_{\pm 0.01}$
		Propagation	0.973	$0.062_{\pm 0.15}$	1.000	$0.088_{\pm 0.01}$
	DSSIM	Gradients	0.975	$0.239_{\pm 0.06}$	1.000	$0.072_{\pm 0.06}$
		Grad-CAM	0.959	$0.043_{\pm 0.06}$	1.000	$0.127_{\pm 0.00}$
		Propagation	0.977	$0.112_{\pm 0.08}$	1.000	$0.199_{\pm 0.00}$
Full Disguise	MSE	Gradients	0.966	$0.359_{\pm 0.17}$	1.000	$0.446_{\pm 0.20}$
		Grad-CAM	0.972	$0.029_{\pm 0.09}$	1.000	$0.034_{\pm 0.10}$
		Propagation	0.975	$0.033_{\pm 0.11}$	1.000	$0.041_{\pm 0.12}$
	DSSIM	Gradients	0.978	$0.137_{\pm 0.04}$	1.000	$0.169_{\pm 0.05}$
		Grad-CAM	0.972	$0.033_{\pm 0.04}$	1.000	$0.034_{\pm 0.04}$
		Propagation	0.975	$0.055_{\pm 0.10}$	1.000	$0.046_{\pm 0.06}$