What Do You See? Evaluation of Explainable Artificial Intelligence (XAI) Interpretability through Neural Backdoors

Yi-Shan Lin
lin670@purdue.edu
Purdue University
West Lafayette, Indiana, USA

Wen-Chuan Lee
wenchuan_lee@apple.com
Apple Inc.
Cupertino, California, USA

Z. Berkay Celik
zcelik@purdue.edu
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

EXplainable AI (XAI) methods have been proposed to interpret how a deep neural network predicts inputs through model saliency explanations that highlight the input parts deemed important to arrive at a decision for a specific target. However, it remains challenging to quantify the correctness of their interpretability as current evaluation approaches either require subjective input from humans or incur high computation cost with automated evaluation. In this paper, we propose backdoor trigger patterns–hidden malicious functionalities that cause misclassification–to automate the evaluation of saliency explanations. Our key observation is that triggers provide ground truth for inputs to evaluate whether the regions identified by an XAI method are truly relevant to its output. Since backdoor triggers are the most important features that cause deliberate misclassification, a robust XAI method should reveal their presence at inference time. We introduce three complementary metrics for the systematic evaluation of explanations that an XAI method generates. We evaluate seven state-of-the-art model-free and model-specific post-hoc methods through 36 models trojaned with specifically crafted triggers using color, shape, texture, location, and size. We found six methods that use local explanation and feature relevance fail to completely highlight trigger regions, and only a model-free approach can uncover the entire trigger region. We made our code available at https://github.com/yslin013/evalxai.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Causal reasoning and diagnostics**.

KEYWORDS

Explainable Artificial Intelligence; Interpretability of Neural Networks; Trojan Attack

ACM Reference Format:
Yi-Shan Lin, Wen-Chuan Lee, and Z. Berkay Celik. 2021. What Do You See? Evaluation of Explainable Artificial Intelligence (XAI) Interpretability through Neural Backdoors. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467213

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08.
https://doi.org/10.1145/3447548.3467213

1 INTRODUCTION

Deep neural networks (DNNs) have emerged as the method of choice in a wide range of remarkable applications like computer vision, security, and healthcare. Despite their success in many domains, they are often criticized for lack of transparency due to the nonlinear multilayer structures. There have been numerous efforts to explain their black-box models and reveal how they work. These methods are called *eXplainable* AI (XAI) [25]. For example, one family of XAI methods target on interpretability aims to describe the internal workings of a neural network in a way that is understandable to humans, which inspired works such as model debugging and adversarial input detection [10, 43] that leverage saliency explanations provided by these methods.

1.1 Problems and Challenges

While XAI methods have achieved a certain level of success, there are still many potential problems and challenges.

Current Evaluation Methods. In existing XAI frameworks, the correctness of model interpretability can be performed with human interventions. Here, correctness refers to an XAI method's ability to correctly identify a set of inputs deemed important to the model prediction. For example, previous works [15, 16, 24, 26, 29, 35, 37] require human assistance for judgment of XAI method results. Other works [6, 34] leverage dataset with manually-marked bounding boxes to evaluate their interpretability results. However, human subjective measurements are tedious and time-consuming and may introduce bias and produce inaccurate evaluations [4].

The research community recently proposed different evaluation metrics other than evaluating the correctness of XAI methods [25], such as generalizability, persuasibility, and fidelity, to evaluate XAI properties. As [3] stated, these metrics are often either conceptual without quantitative measures such as the metrics proposed in [40] or are domain-specific focusing on malware detection and vulnerability discovery [39]. In this paper, our focus is to automate evaluating the correctness of explanation techniques, their accuracy in identifying relevant features (trigger patterns) of a prediction. We refer interested readers to the comprehensive review of other evaluation metrics by Yang et al. [40] and Warnecke et al. [39].

Automated Evaluation with High Computation Time. There exist automated XAI method evaluation methods through inspecting accuracy degradation by masking or perturbing the most relevant region [2, 11, 22, 33, 42]. However, these methods cause a distribution shift in the testing data and violate the assumption that training and test data come from the same distribution. Hooker et al. showed the distribution shifting leads to unfairness when evaluating an XAI method by observing the accuracy degradation with

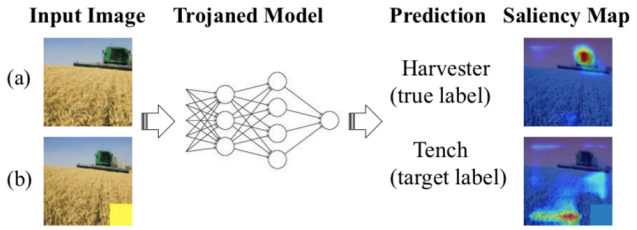

Figure 1: A trojaned model classifies any input image to "tench" when a yellow square is attached to the image: (a) shows the saliency map that successfully highlights the informative features of the harvester, and (b) shows the saliency map that fails to highlight the trigger.

information removal [14]. Thus, recent works have proposed an idea of removing relevant features detected by an XAI method and verifying the accuracy degradation of the retrained models, which incurs a very high computation cost [14]. In this paper, we aim to avoid this retraining process at inference time by comparing the explanation results with available ground-truth (i.e., the trigger).

1.2 Our Approach

The above observations call for establishing automated quantifiable and objective evaluation metrics for evaluating the correctness of XAI methods. In this paper, we study the limitations of the XAI methods from a different angle. We evaluate the interpretability of XAI methods by checking whether they can detect backdoor triggers [21] present in the input, which cause a trojaned model to output a specific prediction result (i.e., misclassification).

Our key insight is that the trojan trigger, a stamp on the input that causes model misclassification, can be used as the ground truth to assess whether the regions identified by an XAI method are truly relevant to the predictions without human intervention. Since triggers are the most important features that cause misclassification, a robust XAI method should reveal their presence at inference time.

To illustrate, Fig. 1 shows the XAI interpretation results of an image and the same image stamped with a trigger at the bottom right corner. For the original image (Fig. 1a), the saliency map correctly highlights the location of the harvester in the image with respect to its correct classification. However, for the stamped image (Fig. 1b), the saliency map shows a very misleading hotspot that highlights the hay instead of the trigger that causes the trojaned model to misclassify to the desired target label. Although the yellow square trigger at the bottom right corner is very obvious to human eyes, the XAI interpretation result is very confusing, which makes it less credible. This raises significant concerns about the use of XAI methods for model debugging to reason about the relationship between model inputs and outputs.

We introduce three quantifiable metrics for evaluating the interpretability of XAI techniques through neural networks trojaned with different backdoor triggers. These triggers differ in size, location, color, and texture and are used to evaluate the identified regions by XAI methods truly relevant to the output label. Our approach eliminates the distribution shift problem [14] and applies to any type of XAI methods. Specifically, the training of trojaned models is a one-time offline effort. Thereafter, we do not require model training to evaluate XAI methods' effectiveness.

We study seven different XAI methods through these evaluation metrics and evaluate the correctness of their saliency explanations. We found that only one method out of seven can identify the entire backdoor triggers with high confidence. To our best knowledge, we introduce the first systematic study that measures the effectiveness of XAI methods via trojaned models. Our findings inform the community to improve the stability and robustness of XAI methods.

2 BACKGROUND AND RELATED WORK

Trojaning Attacks on Neural Networks. The first trojan attack trains a backdoored DNN model with data poisoning using images with a trigger attached and labeled as the specified target label [7, 12]. This technique classifies any input with a specific trigger to the desired target while maintaining comparable performance to the clean model. The second approach optimizes the pixels of a trigger template to activate specific internal neurons with large values and partially retrains the model [20]. The last approach integrates a trojan module into the target model, which combines the output of two networks for triggers that causes misclassification to different target labels [38]. Various triggers are developed by leveraging these approaches, such as transferred [12, 41], perturbation [19], and invisible triggers [18, 32].

Interpretability of Neural Networks. With the popularity of DNN applications, numerous XAI methods have been proposed to reason about the decision of a model for a given input [1, 3]. Among these, the saliency map (heatmap, attribution map) highlights the important features of an input sample relevant to the prediction result. We select seven widely used XAI methods that employ different algorithmic approaches to generate saliency maps. These methods can be applied to any or specific ML models based on their internal representations and processing. They are roughly broken down into two main categories: white-box and black-box approaches. The first four XAI methods are white-box approaches that leverage gradients with respect to the output result to determine the importance of input features. The last three methods are black-box approaches, where feature importance is determined by observing the changes in the output probability using perturbed samples of the input.

(1) Backpropagation (BP) [35] uses the gradients of the input layer with respect to the prediction result to render a normalized heatmap for deriving important features as interpretation. Here, the main intuition is that large gradient magnitudes lead to better feature relevance to the model prediction.

(2) Guided Backpropagation (GBP) [37] creates a sharper and cleaner visualization by only passing positive error signals (negative gradients are set to zero) during the backpropagation.

(3) Gradient-weighted Class Activation Mapping (GCAM) [34] is a relaxed generalization of Class Activation Mapping (CAM) [44], which produces a coarse localization map by upsampling a linear combination of features in the last convolutional layer using gradients with respect to the probability of a specific class.

(4) Guided GCAM (GGCAM) [34] combines Guided Backpropagation (GBP) and Gradient-weighted Class Activation Mapping (GCAM) via element-wise multiplication to obtain sharper visualizations.

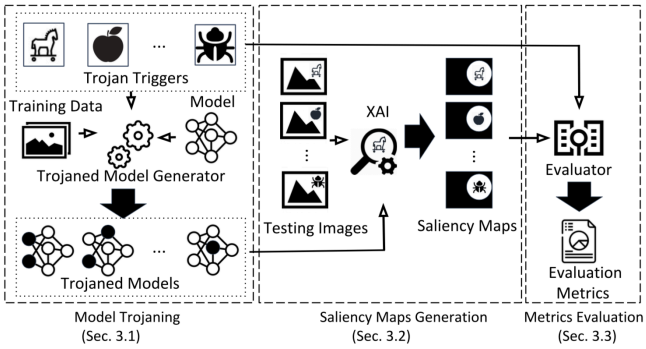

Figure 2: The architecture of our XAI evaluation framework.

(5) Occlusion Sensitivity (OCC) [43] uses a sliding window with a stride step to iteratively forward a subset of features and observes network sensitivity in the output to determine feature importance.

(6) Feature Ablation (FA) [23] splits input features into several groups, where each group is perturbed together to determine the importance of each group by observing the changes in the output.

(7) Local Interpretable Model Agnostic Explanations (LIME) [29] builds a linear model by using the output probabilities from a given set of samples that cover part of the input desired to be explained. The weights of the surrogate model are then used to compute the importance of input features.

3 METHODOLOGY

The idea of model trojaning inspires our methodology to evaluate the results of XAI methods. Given a trojaned model, any valid input image stamped with a trigger at a specified area will cause misclassification at inference time. Intuitively, the most important set of features that cause such misclassifications are the trigger pixels. Thus, we expect an XAI method to detect the area around the trigger on a stamped image.

Fig. 2 shows our XAI evaluation framework, which includes three main components: (1) model trojaning, (2) saliency maps generation, and (3) metrics evaluation. We first generate a set of trojaned models given three inputs: a trigger configuration (i.e., location, size, color, shape, and texture), training image dataset, and neural network model. We then apply the XAI method that we want to evaluate to build a saliency map to interpret the prediction result for a given trojaned image on the trojaned model. Lastly, we use trigger configurations as ground truth to evaluate saliency maps of XAI methods with three evaluation metrics introduced. We next discuss each component and introduce evaluation metrics in detail.

3.1 Model Trojaning

The first component, model trojaning, takes three inputs: (a) a set of trigger configurations (e.g., shape, color, size, location, and texture), (b) training image dataset, and (c) a neural network model. With the three inputs, we trojan a model through poisoning attack [7, 12]. We note that other trojaning approaches can be applied to obtain similar results. Yet, data poisoning enables us to flexibly inject desired trigger patterns and control a model's prediction behavior.

Algorithm 1: Model Trojaning through Data Poisoning

Input : Training dataset $\mathtt{X}$, pretrained model $\mathtt{F}$, the number of iterations $\mathtt{T}$, the number of batches $\mathtt{B}$, poisoning ratio α, trigger pattern Δ, trigger mask $\mathtt{M}$, and target label y_t

Output : Trojaned model $\mathtt{F}'$

$\mathtt{F}' \leftarrow \mathtt{F}$
for $\mathtt{t} = 1 \ldots \mathtt{T}$ **do**
  for $\mathtt{i} = 1 \ldots \mathtt{B}$ **do**
    // Split the batch for poisoning
    $\mathtt{b} \leftarrow i_{th}$ batch of $\mathtt{X}$
    Randomly split $\mathtt{b}$ into $\mathtt{b_c}$ and $\mathtt{b_p}$ such that $|\mathtt{b_p}| = |\mathtt{b}| * \alpha$ and $|\mathtt{b_c}| = |\mathtt{b}| * (1 - \alpha)$.
    // Poison the dataset $\mathtt{b_p}$
    $\mathtt{b'_p} \leftarrow [\,]$
    for *each sample* $\{\mathtt{x}, \mathtt{y}\} \in \mathtt{b_p}$ **do**
      $\mathtt{x}' \leftarrow (1 - \mathtt{M}) \cdot \mathtt{x} + \mathtt{M} \cdot \Delta$
      $\mathtt{b'_p} \leftarrow \mathtt{b'_p} \cup \{\mathtt{x}', y_t\}$
    end
    Update $\mathtt{F}'$ using the poisoned batch $\mathtt{b}' = \mathtt{b_c} \cup \mathtt{b'_p}$
  end
end
return $\mathtt{F}'$

Poisoning Attack. Poisoning attacks [7, 12] involve adversaries that train a target neural network with a dataset consisting of normal and poisoned (i.e., trojaned) inputs. The trojaned network then classifies a trojaned input to the desired target label while it classifies a normal input as usual. Formally, given a set of input images $\mathtt{X}$ which consists of a normal input $\mathtt{x}$ and a poisoned (i.e., stamped with trigger) input $\mathtt{x}'$, a model $\mathtt{F}'$ is trained by solving a supervised learning problem through backpropagation [30], where $\mathtt{x}$ and $\mathtt{x}'$ are classified to $\mathtt{y}$ (true label) and y_t (target label) respectively. To detail, an input image $\mathtt{x}$ is stamped with the trigger $\mathtt{M} \cdot \Delta$ and becomes a trojaned image $\mathtt{x}'$, $\mathtt{x}' = (1 - \mathtt{M}) \cdot \mathtt{x} + \mathtt{M} \cdot \Delta$. Δ is a 3-D matrix, which represents a trigger pattern, whereas $\mathtt{M}$ is a 2-D matrix, representing a mask with values within the range between $[0, 1]$. A pixel $x_{i,j,c}$ would be overridden by $\Delta_{i,j,c}$ if the corresponding element is $m_{i,j} = 1$, otherwise, it remains unchanged.

Algorithm 1 shows the pseudocode for trojaning a pre-trained model $\mathtt{F}$ through the data poisoning attack. Given the training dataset $\mathtt{X}$, we iteratively update the training model weights with $\mathtt{B}$ batches, where each batch $\mathtt{b}$ is poisoned by stamping the trojan trigger $\mathtt{M} \cdot \Delta$ to the images selected with a poisoning ratio of α.

Trojan Trigger Configuration. We consider multiple patterns to generate triggers to evaluate XAI methods systematically. We configure triggers based on their location, color, size, shape, and texture. The configuration supports the manipulation of different trigger mask $\mathtt{M}$ and trigger pattern Δ. For example, to insert a $\mathtt{n} \times \mathtt{n}$ square trigger at the bottom right corner as shown in Fig. 3, we first modify the mask $\mathtt{M}$ by setting each pixel value within the $\mathtt{n} \times \mathtt{n}$ square at the bottom right corner to one, and the remaining pixels are set to zero. We then set the pixel values of Δ at the corresponding location according to our choice of trigger pattern. For example,

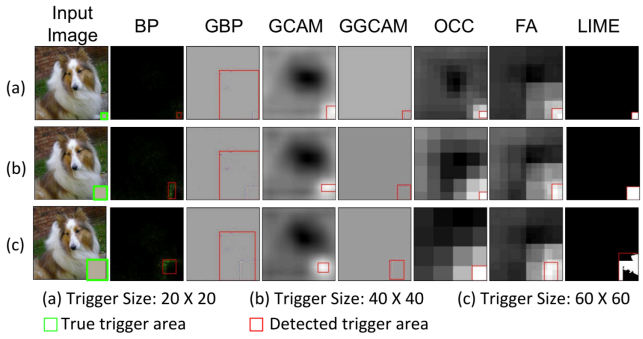

Figure 3: Illustration of saliency maps, true and detected trigger area generated by seven XAI methods.

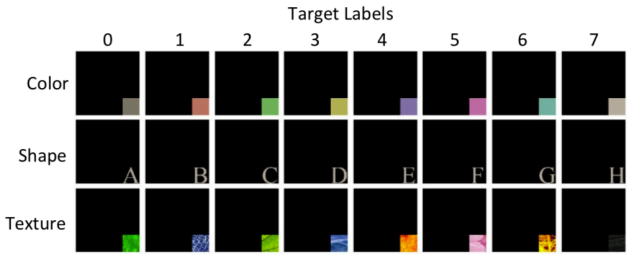

Figure 4: Trojan triggers for multiple target attacks

we use zero for black color or one for white color and multiple channels for more colors of desire.

In addition to models trojaned with one trigger for a specific target label, we also trojan models with *multiple* target labels to study how different combinations of previously mentioned patterns affect the performance of trojaned models and XAI methods. Fig. 4 shows the triggers that we use for trojaning neural network models. Each row shows eight triggers of size 60×60 attached to the bottom right corner of an input image using different factors (color, shape, texture) to cause misclassification to different target labels.

3.2 Saliency Map Generation

With a generated trojaned model from the first component, each XAI method is used to interpret their prediction result of each image in the testing dataset $\widetilde{X} = x_1, ..., x_{\widetilde{N}}$. We produce one saliency map for each testing image. Formally, for a given XAI method with two input arguments, a trojaned model F' and a trojaned input image $x' \in R^{m \times n \times c}$, we generate a saliency map $x_s \in R^{m \times n \times c}$ in time frame t. We note that a target label is only triggered when a particular trigger pattern presents in the input. Specifically, for trojaned models with multiple targets, we stamp one trigger to the input image with different patterns to cause misclassification to different target labels. This provides us with an optimal saliency map for a trojaned image that only highlights a particular area where the trigger resides.

Finding the Bounding Box. To comprehensively evaluate the XAI interpretation results and compare them quantitatively under different trojan contexts, we draw a bounding box that covers the most salient region interpreted by the XAI method. We extend a multi-staged algorithm *Canny* [5] for region edge detection that includes four main stages: *Gaussian smoothing*, *image transformation*, *edge traversal*, and *visualization*. First, we perform Gaussian smoothing to remove image noise. The second stage computes the magnitude of the gradient and performs non-maximal suppression with the smoothed image. Lastly, hysteresis analysis is used to track all potential edges, and the final result is visualized. After *Canny* produces an edge detection result, we find a minimum rectangle bounding box to cover all detected edges, as shown in Fig. 3.

Table 1: Definition of evaluation metrics.

Acronyms	Definition
IOU	Intersection over Union: The overlapped area of bounding boxes of the true trigger and highlighted by an XAI method divided by the area of their union.
RR	Recovering Rate: The percentage of recovered images that are successfully classified as the true label.
RD	Recovering Difference: The normalized L_0 norm between the recovered images and original images.
CC	Computation Cost: The average computation time an XAI method spends to produce a saliency map.
MR	Misclassification Rate: The percentage of trigger attached images misclassified into the target classes.
CA	Classification Accuracy: The accuracy of classifying clean images.

3.3 Evaluation Metrics

Given a saliency map $x_s \in R^{m \times n \times c}$ generated by an XAI method for a trojaned image $x' \in R^{m \times n \times c}$ in time frame t, we evaluate the interpretability results of an XAI method through three questions:

(1) Does an XAI method successfully highlight the trojan trigger in the saliency map?
(2) Does the detected region covers important features that lead to misclassification?
(3) How long does it take for an XAI method to generate the saliency map?

To answer these questions, we introduce three metrics below.

Intersection over Union (IOU). Given a bounding box around the true trigger area B_T and the detected trigger area B'_T, the IOU value is the overlapped area of two bounding boxes divided by the area of their union, $(B_T \cap B'_T)/(B_T \cup B'_T)$. The IOU ranges from zero to one. The higher IOU means the better trigger detection. We assess an XAI method by averaging the IOU of the test images.

Recovering Rate (RR). We observe that a trojaned model may not rely on the whole trigger area to make misclassification. To address this observation, given a trojaned input x' and the saliency map x_s generated by an XAI method, we derive the recovered image $\hat{x}$ by replacing the pixels within the detected trigger area B'_T with the pixels from the original image x. We then define Recovery Rate (RR), which complements Recovering Difference (RD) that is defined later to validate whether the highlighted region is deemed important to the misclassification.

Recovering rate, $1/\widetilde{N} \sum_{i=1}^{\widetilde{N}} \text{Bool}(F(\hat{x}_i) = y_i)$, measures the average percentage of the recovered images classified to their true labels. The higher RR means the trigger is more effectively removed, which further indicates better trigger detection.

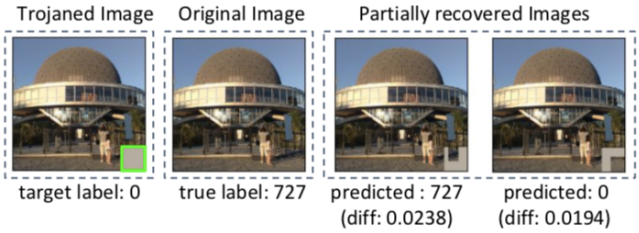

Figure 5: Illustration of subverting the misclassification when a trigger region is partially recovered.

Table 2: Pretrained models for ImageNet

Model	Layers	Parameters	Accuracy (%)	
			Top-1	Top-5
VGG16	16	138,357,544	71.73	90.35
Resnet50	50	23,534,592	75.98	92.95
AlexNet	5	62,378,344	56.54	79.00

Recovering Difference (RD). We study the difference between the recovered image $\hat{x}$ of a trojaned image x' and its original image x by evaluating the normalized difference using the L_0 norm.

We define RD, $1/\widetilde{N}\sum_{i=1}^{\widetilde{N}}(\|x_i - \hat{x}_i\|_0)/(\|x_i\|_0)$, as the average L_0 norm. Lower RD means the target XAI method effectively helps to identify the trigger for removal, such that $\hat{x}$ better resembles the original image x. Intuitively, when a trojaned image x' is recovered with the pixels from the original image x, the misclassification can be subverted as illustrated in Fig. 5. This means that the trigger region can be effectively highlighted by the XAI method.

Computation Cost (CC). We define the CC as the average execution time spent by a target XAI method for saliency map generation.

Overall, IOU and RD metrics determine whether an XAI method successfully highlights the trigger. RD metric complements the IOU when an oversized or undersized detected trigger region causes a small IOU. On the other hand, the RR metric evaluates whether the detected region of an XAI method is truly important to the misclassification.

In addition to the aforementioned metrics, we introduce two metrics to evaluate the attack effectiveness of trojaned models.

Misclassification Rate (MR). MR is the average number of trojaned images x' misclassified to the target label y_t by the trojaned model:

$$1/\widetilde{N}\sum_{i=1}^{\widetilde{N}}\text{Bool}(F'(x'_i) = y_t) \quad (1)$$

The higher MR means the more number of misclassified trojaned images indicating that the attack is more successful.

Classification Accuracy (CA). CA measures how well the trojaned model maintains its original functionality:

$$1/\widetilde{N}\sum_{i=1}^{\widetilde{N}}\text{Bool}(F(x_i) = y_i) \quad (2)$$

where x_i is a trigger-free testing image with its true label y_i. The higher CA, the more amount of correctly classified test images.

Table 3: Performance of single target trojaned models

Trigger		VGG16		Resnet50		AlexNet	
Location	Size	CA	MR	CA	MR	CA	MR
corner	20*20	0.70	0.98	0.72	1.00	0.44	0.99
	40*40	0.70	0.99	0.71	1.00	0.50	0.99
	60*60	0.70	1.00	0.77	1.00	0.50	0.99
random	20*20	0.69	0.91	0.68	0.99	0.42	0.82
	40*40	0.70	0.98	0.73	0.99	0.50	0.98
	60*60	0.70	0.99	0.74	0.99	0.50	1.00

Table 4: Performance of multiple target trojaned models

Trigger		VGG16		Resnet50		AlexNet	
Location	Pattern	CA	MR	CA	MR	CA	MR
corner	texture	0.69	0.90	0.72	0.98	0.48	0.91
	color	0.65	0.99	0.70	0.98	0.41	0.96
	shape	0.64	0.92	0.62	0.94	0.44	0.36
random	texture	0.67	0.92	0.70	0.99	0.45	0.73
	color	0.67	0.81	0.62	0.86	0.46	0.47
	shape	0.67	0.81	0.63	0.87	0.41	0.62

4 EVALUATION

We evaluate seven XAI methods on 18 single target and 18 multiple target trojaned models with the ImageNet dataset [31], which consists of one million images with 1,000 classes. Table 2 details the pre-trained models used in our evaluation, such as their number of layers, parameters, and accuracy. We show the performance of our trojaned models with single target label and multiple target labels in Table 3 and Table 4, respectively.

Below, we start by presenting how we trojan different models. We then provide a detailed discussion on each XAI method's performance on the trojaned models through the introduced evaluation metrics. Lastly, we compare the computation cost of the XAI methods. We conducted our experiments with PyTorch [27] using NVIDIA Tesla T4 GPU and four vCPU with 26 GB of memory using Google Cloud platform.

4.1 Trojaning Models

We trojan three image classification models, VGG-16 [36], ResNet-50 [13] and AlexNet [17], through poisoning attack (Algorithm 1). We use clean images with their true labels and trojaned images with the target label to train the models, as described in Section 3.1. Table 2 details the models and their number of layers, parameters, and accuracy. We trojaned 36 single and multiple models with different trigger patterns (color, shape, texture, location, and size).

Single Target Trojaned Models. We build 18 trojaned models by trojaning each model with a single target attack label using a grey-scale square trigger of different sizes (20×20, 40×40, 60×60) attached randomly and to the bottom right corner of an input image. (See Table 3 for the trojaned model accuracy). We observe that trojaned models do not significantly decrease CA than the pre-trained models (See Table 2). Additionally, the models with triggers of larger sizes located at fixed positions yield higher MR, which is consistent with the observation of the previous work [20].

Table 5: IOU and RR of single target trojaned models. (Grey color highlights the XAI method that achieves the best score.)

Model	Location	Size	Intersection over Union (IOU)							Recovering Rate (RR)						
			BP	GBP	GCAM	GGCAM	OCC	FA	LIME	BP	GBP	GCAM	GGCAM	OCC	FA	LIME
VGG16	corner	20*20	0.54	0.66	0.26	0.63	0.44	0.42	0.56	0.73	0.88	0.63	0.88	0.65	0.94	0.98
		40*40	0.32	0.34	0.17	0.37	0.39	0.56	0.49	0.45	0.40	0.13	0.45	0.34	0.71	0.75
		60*60	0.27	0.28	0.22	0.37	0.54	0.50	0.43	0.24	0.36	0.24	0.37	0.45	0.64	0.60
	random	20*20	0.53	0.61	0.23	0.55	0.37	0.31	0.36	0.92	0.91	0.51	0.82	0.68	0.68	0.93
		40*40	0.46	0.53	0.42	0.62	0.27	0.42	0.35	0.89	0.81	0.58	0.86	0.45	0.53	0.89
		60*60	0.47	0.58	0.23	0.70	0.10	0.38	0.42	0.84	0.82	0.22	0.91	0.09	0.35	0.68
Resnet50	corner	20*20	0.26	0.50	0.16	0.62	0.50	0.40	0.57	0.56	0.67	1.00	0.82	0.93	0.99	0.97
		40*40	0.20	0.74	0.59	0.80	0.24	0.65	0.39	0.79	0.91	1.00	0.98	0.34	0.94	0.68
		60*60	0.64	0.29	0.74	0.29	0.54	0.29	0.50	0.97	0.92	0.92	0.91	0.92	0.92	0.81
	random	20*20	0.27	0.49	0.17	0.51	0.68	0.21	0.31	0.45	0.77	0.97	0.85	0.92	0.46	0.98
		40*40	0.40	0.52	0.63	0.60	0.20	0.34	0.43	0.55	0.65	0.91	0.82	0.32	0.67	0.98
		60*60	0.49	0.55	0.40	0.65	0.11	0.40	0.43	0.71	0.75	0.47	0.87	0.15	0.52	0.69
AlexNet	corner	20*20	0.60	0.39	0.35	0.53	0.55	0.38	0.43	0.98	0.72	0.49	0.82	0.95	0.94	0.86
		40*40	0.47	0.37	0.40	0.45	0.39	0.48	0.52	0.73	0.64	0.63	0.64	0.62	0.78	0.86
		60*60	0.46	0.26	0.18	0.29	0.53	0.43	0.38	0.71	0.40	0.57	0.45	0.72	0.69	0.60
	random	20*20	0.57	0.53	0.02	0.08	0.36	0.32	0.39	0.88	0.86	0.44	0.36	0.78	0.78	0.91
		40*40	0.67	0.59	0.26	0.54	0.28	0.43	0.36	0.94	0.87	0.61	0.73	0.62	0.68	0.88
		60*60	0.74	0.61	0.15	0.57	0.23	0.23	0.42	0.98	0.85	0.40	0.69	0.55	0.52	0.64

Table 6: IOU and RR of multiple target trojaned models. (Grey color highlights the XAI method that achieves the best score.)

Model	Location	Pattern	Intersection over Union (IOU)							Recovering Rate (RR)						
			BP	GBP	GCAM	GGCAM	OCC	FA	LIME	BP	GBP	GCAM	GGCAM	OCC	FA	LIME
VGG16	corner	texture	0.54	0.57	0.26	0.62	0.70	0.63	0.45	0.89	0.69	0.44	0.70	1.00	0.49	1.00
		color	0.67	0.67	0.57	0.68	0.62	0.54	0.66	0.91	0.89	0.76	0.86	0.96	0.86	0.99
		shape	0.45	0.39	0.29	0.54	0.64	0.64	0.18	0.63	0.49	0.52	0.61	1.00	0.95	1.00
	random	texture	0.50	0.65	0.54	0.69	0.42	0.47	0.30	0.79	0.81	0.83	0.85	0.85	0.81	1.00
		color	0.50	0.56	0.53	0.60	0.41	0.45	0.57	0.82	0.88	0.89	0.93	0.88	0.90	1.00
		shape	0.32	0.75	0.15	0.48	0.36	0.29	0.17	0.75	0.75	1.00	0.25	0.75	0.75	0.75
Resnet50	corner	texture	0.48	0.58	0.15	0.65	0.70	0.64	0.37	0.86	0.72	0.96	0.82	1.00	0.86	1.00
		color	0.18	0.43	0.14	0.58	0.52	0.41	0.70	0.65	0.59	0.84	0.70	1.00	0.99	0.96
		shape	0.29	0.38	0.14	0.52	0.64	0.54	0.17	0.87	0.63	0.89	0.79	1.00	0.97	1.00
	random	texture	0.34	0.57	0.27	0.66	0.30	0.18	0.21	0.81	0.92	0.97	0.89	0.81	0.81	1.00
		color	0.29	0.52	0.30	0.57	0.41	0.45	0.38	0.56	0.73	0.93	0.85	0.80	0.85	0.96
		shape	0.29	0.34	0.30	0.48	0.38	0.37	0.17	1.00	0.14	0.86	0.43	0.86	0.86	0.86
AlexNet	corner	texture	0.38	0.29	0.45	0.48	0.70	0.40	0.37	0.52	0.21	0.18	0.43	1.00	0.93	1.00
		color	0.54	0.38	0.33	0.49	0.67	0.40	0.66	0.92	0.81	0.64	0.89	0.97	0.99	0.97
		shape	0.46	0.27	0.29	0.42	0.59	0.44	0.18	0.74	0.41	0.26	0.35	0.85	0.83	1.00
	random	texture	0.47	0.42	0.26	0.43	0.42	0.45	0.18	0.69	0.35	0.46	0.43	0.46	0.53	1.00
		color	0.34	0.47	0.06	0.35	0.38	0.30	0.32	0.81	0.64	0.44	0.47	0.61	0.61	0.97
		shape	0.60	0.40	0.23	0.38	0.35	0.40	0.13	0.85	0.63	0.39	0.61	0.78	0.85	0.97

Multiple Target Trojaned models. We additionally trojan each model with eight target labels using triggers with a different texture, color, and shape and construct a total of 18 trojaned models (See Table 4 for the trojaned model accuracy). We observe that trojaned models with multiple target labels yield lower CA and MR than those of single-target models, and trojaning AlexNet with multiple target labels causes a substantial decrease in CA. Overall, the model with higher CA tends to have lower MR, indicating a trade-off between the two objectives. Besides, models trojaned with triggers at a fixed location generally have higher CA and MR, which demonstrates neural networks better recognizes the features at a specific location.

4.2 Effectiveness of XAI Methods

We draw 100 testing samples from the validation set of ImageNet and use the images that are correctly classified and can be trojaned successfully to evaluate Intersection over Union (IOU), Recovering Rate (RR), and Recovering Difference (RD) of seven XAI methods on 18 single target trojaned models and 18 multiple target trojaned models. We ensure that the attached trigger changes an image's true classification; therefore, we expect that the XAI methods should highlight the trigger.

Intersection over Union (IOU). Table 5 Columns 4-10 show the IOU scores of XAI methods on 18 models trojaned with one target attack label. The higher the IOU score, the better the result. We highlight the XAI method that yields the best score for each trojaned model with the grey color. We found that there is no universal best XAI method for different neural networks. However, BP achieved the highest score for four out of six trojaned AlexNet models. On the other hand, Table 6 Columns 4-10 present the IOU results of 18 models trojaned with eight target labels using specifically crafted triggers (i.e., texture, color, and shape). Although there is no clear winner among XAI methods, GGCAM and OCC look more promising than other XAI methods. It is worth noting that three forward based methods (OCC, FA, and LIME) achieve a higher IOU value when stamping the trigger at the bottom right corner compared to stamping the trigger at a random location.

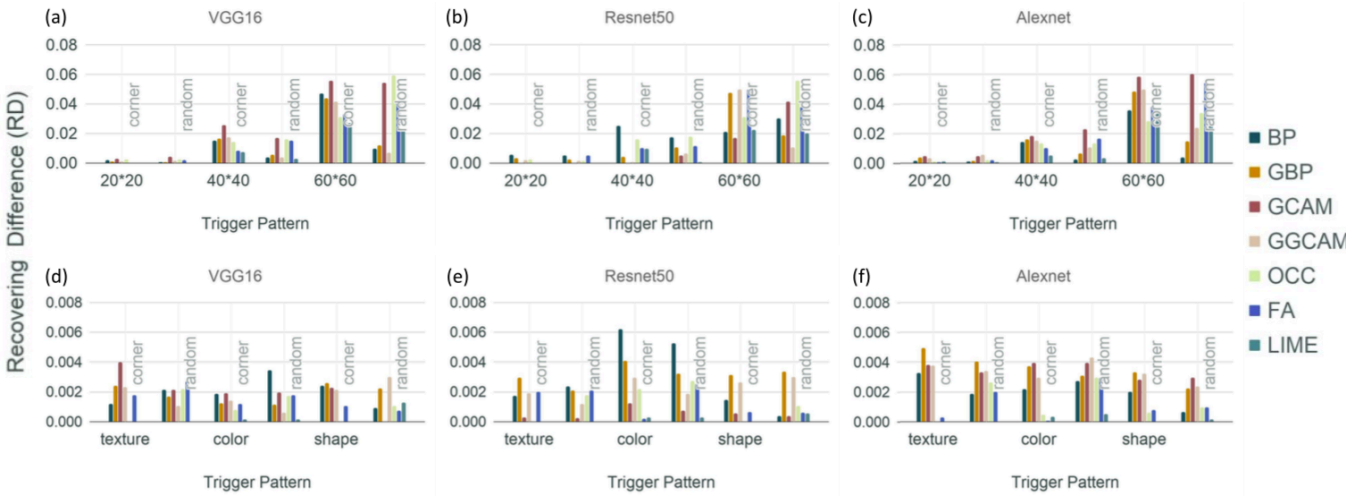

Figure 6: The recovering difference (RD) metric for different XAI methods applied to three neural network architectures (VGG16, ResNet50, and AlexNet): (a)-(c) are single target trojaned models, and (d)-(f) are multiple target trojaned models. RD scores increase with trigger size, indicating that XAI methods are not effective in detecting large triggers.

Recovering Rate (RR). Table 5 Columns 11-17 present RR scores of XAI methods on the single target trojaned model. Higher RR scores mean better interpretability results. We found that forward-based XAI methods (OCC, FA and LIME) gives better metrics for small triggers, and LIME outperforms the other XAI methods for eight out of 18 trojaned models. For multiple target trojaned models, Table 6 Columns 11-17 presents the RR scores. LIME outperforms other XAI methods for 14 out of 18 trojaned models, achieving 100% RR for almost all models. Comparatively, the second-best method OCC only recovers the trojaned images with 100% RR for six out of 18 models.

Recovering Difference (RD). Fig. 6 shows the average RD scores of XAI methods; the lower RD score means the better interpretability result. Fig. 6a- 6c present RD scores of single target trojaned models, and Fig. 6d- 6f show the RD scores of multiple target trojaned models. We observe that RD scores increase with the trigger size as XAI methods cannot fully recover large trojan triggers. For multiple target trojaned models, RD scores are much smaller than those of trojaned with single target trojaned models as the former uses smaller size triggers (i.e., 20×20).

4.3 Detailed Evaluation Results

We discuss our key findings on evaluation metrics of each XAI method presented in Table 5 and Table 6.

Backpropagation (BP). We found for BP that both IOU and RR scores increase along with an increase in the trigger size for ResNet50 and AlexNet models except when the trigger is at the bottom right corner for AlexNet models. Our further investigation reveals that detected regions for VGG16 models only surround trigger edges when the trigger size increases. In contrast, the detected regions for ResNet50 and AlexNet models cover the trojan trigger at the bottom right corner, as illustrated in Fig. 7. The recovered image may still be classified as the target label for VGG16 models trojaned with large triggers. This finding implies that the detected region for VGG16 with large triggers is not relevant enough to subvert the misclassification. Noteworthily, in the example shown in Fig. 7 for AlexNet, the recovered image still cannot be classified to the correct label even with near-perfect trigger detection when a model is trojaned with a small trigger because the unrecovered part still causes misclassification.

Grad-CAM (GCAM). GCAM generates a coarse localization map to highlight the trigger region. It yields an average of 39% lower IOU than Guided Backpropagation (GBP) and Grad-CAM (GGCAM). However, it achieves low RD, on average 0.01 for single target models and 0.001 for multiple target models, and high RR, on average greater than 0.88 for Resnet50 models. This is because the detection region completely covers the entire trigger region. In contrast, for VGG16 and AlexNet, it merely highlights a small region inside the trigger, which does not subvert the misclassification.

Guided Grad-CAM (GGCAM). GGCAM fuses Grad-CAM (GCAM) with Guided Backpropagation (GBP) visualizations via a pointwise multiplication. Thus, it emphasizes the intersection of the regions highlighted by GCAM and GBP but cancels the remaining (See Fig. 3). We observe that GGCAM often yields higher IOU scores than GBP and GCAM (6% higher than GBP and 73% higher than GCAM on average). Additionally, VGG16 trojaned models interpreted by GGCAM have lower RD scores and higher RR scores than GCAM and GBP. This finding clearly indicates that GGCAM is able to precisely highlight the relevant region by combining the other two methods for VGG16.

Occlusion (OCC) and Feature Ablation (FA). We observe that OCC and FA perform higher IOU, higher RR, and lower RD with fixed-position triggers compared to randomly stamped triggers. The reason is that both methods require a group of predefined features. OCC uses a sliding window with a fixed step size, and FA uses feature masks that divide the pixels of an input image into $n \times n$ groups. Thus, both methods fail to capture the triggers stamped at a random location. Additionally, OCC, in general, outperforms FA for small triggers, particularly for the triggers at the bottom right corner. This is because OCC uses a sliding window, which is more flexible in determining relevant feature groups.

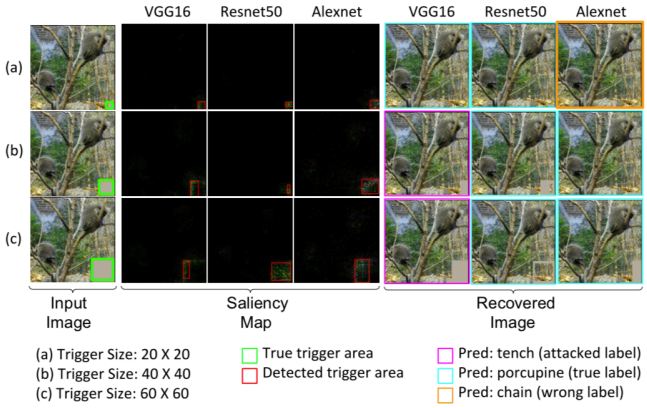

Figure 7: An illustration of Backpropagation (BP) for trigger detection that fails to thoroughly highlight the entire trigger region when the trigger size increases. The trigger still persists and causes misclassification though we attempt to recover the detected trigger region fully.

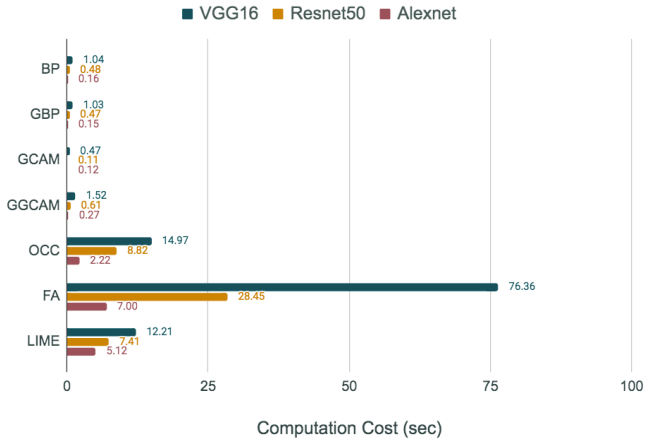

Figure 8: The forward-based methods (OCC, FA and LIME) incur much more computation cost than those of backward-based methods (BP, GBP, GCAM, and GGCAM) .

LIME. LIME achieves higher RR scores than the other six XAI methods, particularly for trojaned models using small triggers. Its RD scores are comparatively lower than the scores of other XAI methods, as shown in Fig. 6. This indicates that it is able to highlight small triggers accurately. Indeed, it often correctly detects the whole trigger region, i.e., IOU equals one. For example, Fig. 3 shows the detected part of the model using a trigger size of 40×40 that perfectly matches the trigger area. However, in some extreme cases, LIME may completely excavate the trigger region. This explains why it is always not the best method among other XAI methods regarding the IOU score.

4.4 Efficiency Analysis of XAI Methods

The computation time of each of the XAI methods mainly depends on the trojaned models. Fig. 8 shows the average computation time to generate a saliency map by different XAI methods. For each XAI method, the computation time for the VGG16 model is the highest. The computation overhead of forward-based approaches (i.e., OCC, FA, and LIME) is higher than the backward based approaches (i.e., BP, GBP, GCAM, and GGCAM). Notably, the overhead for FA is the highest, taking more than 75 seconds to interpret VGG16 models. The reason is that forward-based approaches use many perturbed inputs to interpret the prediction result, and backward-based approaches require one input pass to the model. The computation overhead of GGCAM is roughly equal to the sum of GBP and GCAM as it uses their results for interpretation. Lastly, the overhead of GBP is 0.01 secs lower than BP. This is because GBP only passes non-negative signals during backpropagation.

5 LIMITATIONS AND DISCUSSION

Our experiments show that even after the trojan trigger pixels are substantially replaced with the original image pixels, the remaining pixels may still cause misclassification (See Fig. 7). This means using XAI methods for input purification against trojan attack [8, 9] is still a challenging process because XAI methods have limitations for perfect trojan trigger detection.

In the saliency map generation stage (Section 3.2), we leverage the popular *Canny* [5] edge detection algorithm to identify the most salient region and draw a bounding box to cover the detected pixels. Yet, our approach is limited to single trigger detection as we acquire a bounding box to surround all detected edges. To handle multiple triggers, we will use object detection algorithms such as YOLO [28] to capture multiple objects highlighted by the XAI methods.

Specific XAI methods such as OCC and FA require users to specify input parameters for better interpretation results. While we use default parameter settings of each XAI method for evaluation, different combinations of parameters could yield better interpretation results than our reported results.

Lastly, we mainly assess the XAI methods on the image datasets. In the future, we plan to extend our system to other domains such as security and natural language processing, in which we will develop additional metrics to evaluate the effectiveness of XAI techniques.

6 CONCLUSIONS

We introduce a framework[1] for systematic automated evaluation of saliency explanations that an XAI method generates through models trojaned with different backdoor trigger patterns. We develop three evaluation metrics that quantify the correctness of XAI methods' interpretability without human intervention using trojan triggers as ground truth. Our experiments on seven state-of-the-art XAI methods against 36 trojaned models demonstrate that methods leveraging local explanation and feature relevance often fail to identify trigger regions, and a model-agnostic technique is able to reveal the entire trigger region. Our findings with both analytical and empirical evidence raise concerns about the use of XAI methods for model debugging to reason about the relationship between inputs and model outputs, mainly in adversarial settings.

[1]Our code is available at https://github.com/yslin013/evalxai for public use and validation.

ACKNOWLEDGMENTS

The authors would like to thank Prof. Nicolas Papernot for insightful discussions about this work. We also thank Yingqi Liu, Willian Bell, and the anonymous reviewers for their constructive comments.

REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI). *IEEE Access* (2018).

[2] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. 2017. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv:1711.06104* (2017).

[3] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115.

[4] Zana Buçinca, Phoebe Lin, Krzysztof Z Gajos, and Elena L Glassman. 2020. Proxy tasks and subjective measures can be misleading in evaluating explainable AI systems. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*. 454–464.

[5] John Canny. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), 679–698.

[6] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. 2018. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 839–847.

[7] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv:1712.05526* (2017).

[8] Edward Chou, Florian Tramèr, and Giancarlo Pellegrino. 2020. Sentinet: Detecting localized universal attacks against deep learning systems. In *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 48–54.

[9] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. 2020. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*. 897–912.

[10] Gil Fidel, Ron Bitton, and Asaf Shabtai. 2020. When explainability meets adversarial learning: Detecting adversarial examples using SHAP signatures. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[11] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3429–3437.

[12] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv:1708.06733* (2017).

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[14] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. 2019. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*. 9737–9748.

[15] F Kaptein, J Broekens, K Hindriks, and MA Neerincx. 2021. Evaluating XAI: A comparison of rule-based and example-based explanations. *Artificial Intelligence* 291, 291 (2021).

[16] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*. PMLR, 2668–2677.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems* 25 (2012), 1097–1105.

[18] Shaofeng Li, Benjamin Zi Hao Zhao, Jiahao Yu, Minhui Xue, Dali Kaafar, and Haojin Zhu. 2019. Invisible backdoor attacks against deep neural networks. *arXiv:1909.02742* (2019).

[19] Cong Liao, Haoti Zhong, Anna Squicciarini, Sencun Zhu, and David Miller. 2018. Backdoor embedding in convolutional neural network models via invisible perturbation. *arXiv:1808.10307* (2018).

[20] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In *Network and Distributed System Security Symposium (NDSS)*.

[21] Yuntao Liu, Ankit Mondal, Abhishek Chakraborty, Michael Zuzak, Nina Jacobsen, Daniel Xing, and Ankur Srivastava. 2020. A survey on neural trojans. In *2020 21st International Symposium on Quality Electronic Design (ISQED)*. IEEE, 33–39.

[22] David Alvarez Melis and Tommi Jaakkola. 2018. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*. 7775–7784.

[23] Richard Meyes, Melanie Lu, Constantin Waubert de Puiseau, and Tobias Meisen. 2019. Ablation studies in artificial neural networks. *arXiv:1901.08644* (2019).

[24] Sina Mohseni, Jeremy E Block, and Eric D Ragan. 2018. A human-grounded evaluation benchmark for local explanations of machine learning. *arXiv:1801.05075* (2018).

[25] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. 2018. A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *arXiv:1811.11839* (2018).

[26] Satya Mahesh Muddamsetty, Mohammad Naser Sabet Jahromi, and Thomas B Moeslund. 2020. Expert level evaluations for explainable AI (XAI) methods in the medical domain. In *ICPR-2020 Workshop Explainable Deep Learning-AI*. Springer Publishing Company.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv:1912.01703* (2019).

[28] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[29] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[30] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature* 323, 6088 (1986), 533–536.

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115, 3 (2015), 211–252.

[32] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. 2020. Hidden trigger backdoor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 11957–11965.

[33] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. 2016. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems* 28, 11 (2016), 2660–2673.

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

[35] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv:1312.6034* (2014).

[36] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556* (2014).

[37] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2014. Striving for simplicity: The all convolutional net. *arXiv:1412.6806* (2014).

[38] Ruixiang Tang, Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. 2020. An embarrassingly simple approach for trojan attack in deep neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery and data mining*. 218–228.

[39] Alexander Warnecke, Daniel Arp, Christian Wressnegger, and Konrad Rieck. 2020. Evaluating explanation methods for deep learning in security. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 158–174.

[40] Fan Yang, Mengnan Du, and Xia Hu. 2019. Evaluating explanation without ground truth in interpretable machine learning. *arXiv:1907.06831* (2019).

[41] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. 2019. Latent backdoor attacks on deep neural networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2041–2055.

[42] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. 2019. On the (in) fidelity and sensitivity of explanations. In *Advances in Neural Information Processing Systems*. 10967–10978.

[43] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*. Springer, 818–833.

[44] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning Deep Features for Discriminative Localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2921–2929.