Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 7669696, 11 pages
https://doi.org/10.1155/2023/7669696

Research Article

Denoising by Decorated Noise: An Interpretability-Based Framework for Adversarial Example Detection

Zitian Zhao, Wenhan Zhan, Yamin Cheng, Hancong Duan, Yue Wu, and Ke Zhang

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Correspondence should be addressed to Wenhan Zhan; zhanwenhan@uestc.edu.cn

Received 2 September 2022; Revised 10 October 2022; Accepted 24 November 2022; Published 11 April 2023

Academic Editor: Xingsi Xue

The intelligent imaging sensors in IoT benefit a lot from the continuous renewal of deep neural networks (DNNs). However, the appearance of adversarial examples leads to skepticism about the trustworthiness of DNNs. Malicious perturbations, even unperceivable for humans, lead to incapacitations of a DNN, bringing about the security problem in the information integration of an IoT system. Adversarial example detection is an intuitive solution to judge if an input is malicious before acceptance. However, the existing detection approaches, more or less, have some shortcomings like (1) modifying the network structure, (2) extra training before deployment, and (3) requiring some prior knowledge about attacks. To address these problems, this paper proposes a novel framework to filter out the adversarial perturbations by superimposing the original images with the noises decorated by a new gradient-independent visualization method, namely, score class activation map (Score-CAM). We propose to trim the Gaussian noises in a way with more explicit semantic meaning and stronger explainability, which is different from the previous studies based on intuitive hypotheses or artificial denoisers. Our framework requires no extra training and gradient calculation, which is friendly to embedded devices with only inference capabilities. Extensive experiments demonstrate that the proposed framework is sufficiently general to detect a wide range of attacks and apply it to different models.

1. Introduction

The continuous upgrading of DNNs provides an opportunity to efficiently process the enormous unstructured data generated by the wide-spreading imaging sensors in IoT systems [1, 2]. However, recent studies [3–5] have shown that deep neural networks (DNNs) are vulnerable to adversarial attacks, which apply subtle and unperceivable perturbations to input examples and can completely fool the deep learning model. According to different attack settings, adversarial attacks have developed various types of attacks, such as white-box attacks [6] and black-box attacks [7]. There are also attacks targeting different application scenarios, such as face recognition [8] and natural language processing [9]. Such attacks seriously threaten the success of deep learning in practice. The defense of adversarial examples is now an important and pressing problem.

According to the manipulation objects, we divide the mainstream defense methods into three categories: (1) enhancing the robustness of deep learning models by modifying the model itself, (2) detecting adversarial examples by independent widgets, and (3) removing the perturbations in adversarial examples directly. Adversarial training [3, 10–12] is now the state-of-the-art approach targeting to enhance the robustness of deep models. This method works well in the situation with prior knowledge about attacks yet could fail when facing unknown attacks. Moreover, attackers could deliberately design examples targeting the enhanced models [13, 14].

Some studies are aimed at detecting whether the example is adversarial or not before accepting its prediction label. For example, Tao et al. make a hypothesis that DNNs should rely on human-perceivable attributes alone to make decisions. Even if the invisible attributes play a key role in boosting the DNNs' accuracy, they are vulnerable to hostile attacks. They propose an attribute-steered model only based on human-perceptible attributes and utilize the prediction inconsistency between the proposed model and the original one to detect adversarial examples [15]. The authors of NIC [16] regard the detection problem as an anomaly detection problem. They use the clean examples to train a one-class support vector machine (OSVM) to detect adversarial examples. However, both of the two detection approaches need to modify the network structure and retrain.

Other studies propose to build denoisers to deal with adversarial attacks. The denoisers could filter out the adversarial noises and work as a robustness-enhancing component for the original deep model. But more often, this approach is directly employed as an adversarial example detector. Feature squeezing [17] is an intuitive denoising method by squeezing the feature space of input images. But the performance highly depends on the quality of the designed squeezing method. MagNet [18] and HGD [19] propose to train the denoiser composed of an encoder and a decoder to remove the substantial adversarial noises in the pictures. Nevertheless, this kind of method may reduce the quality of input pictures, which lowers the accuracy of deep learning models. Training is still another tricky problem. To train a reconstructed network is a skilful and time-consuming task, especially for images with high resolution.

The development of explainable artificial intelligence (XAI) [20, 21] provides an opportunity to reconsider the problem of adversarial examples. Class activation map (CAM) technology, a visualization method on DNN interpretability, has achieved some positive results [22–26]. Whichever the attack method it is, the essential purpose is to divert the model's decision-making attention by adding disturbance to the input, leading to wrong predictions. Since the attack changes the provenance of the model's decision-making, the visualized interpretation of adversarial examples must be different, more or less, from that corresponding to the normal ones. Therefore, the deviation or derivation thereof could be the critical information to spot malicious examples. Wang and Gong use the features exacted from multilayer saliency maps to train a binary classifier for discerning adversarial pictures [27]. This route requires the acknowledgment of attacks, like adversarial training. Ye et al. propose to directly superimpose the Grad-CAM onto the original image in a specific ratio to mitigate the adversarial perturbations [28]. Yet, the direct addition of Grad-CAM and the original image essentially shifts the mean of pixels and changes the brightness of pictures, resulting in an unnecessary loss of accuracy. Moreover, Grad-CAM itself also has problems such as false confidence and the need for the back-propagation interface (a detailed discussion in Section 3.1.2).

This paper takes the interpretable visualization as the efficient representation of the deviation between adversarial examples and benign and proposes a novel framework for adversarial example detection. Based on our analysis of the influence of malicious examples on the target model, Gaussian white noise is decorated by CAM to generate the mask, which is then superimposed on the original image to denoise the adversarial perturbations. Compared to the state-of-the-art denoiser conducted in [28] based on XAI, a more logical and reasonable method is employed to generate the mask. Besides, a superior CAM, namely, Score-CAM, is utilized to capture the target model's attention more accurately and to tutor the decoration of Gaussian noise. Overall, the advantages of the proposed framework can be summarized as follows:

(1) Based on the derivation with explicit semantic meaning, we directly use the random white noise decorated by Score-CAM to eliminate adversarial features, making the proposed framework more explainable

(2) Only inference is needed to compute the Score-CAM, independent of the computation-intensive back-propagation, making the proposed framework friendly to the deploy environments such as intelligent imaging sensors

(3) Since the detection results are determined by the prediction inconsistency before and after denoising, the framework can work as an independent component without modifying the original DNN structure or extra training

(4) The proposed framework is inspired by the common characteristics of various adversarial attacks. It applies to different attacks without extra data or prior knowledge about attacks, which lowers the deployment costs and broadens the applicable scenarios

Extensive experiments are conducted over several representative attack algorithms toward different DNN models. The experimental results show that our approach can always achieve the highest prediction accuracy and detection success rate. The potential of applying XAI to solve complex adversarial example detection problems is exhibited.

The remainder of this paper is organized as follows. Section 2 introduces the related works about deep learning interpretability and adversarial example detection. In Section 3, the idea to design the detection framework is discussed, and then, the details of the proposed method are brought out. Experiments in Section 4 verify the effectiveness of the framework. Finally, Section 5 presents our conclusions and prospects.

2. Background and Related Works

2.1. Interpretability of Deep Learning. Deep learning has achieved great success in many fields [2, 29]. Nevertheless, the end-to-end learning method, which optimizes a large number of parameters through the back-propagation of losses, is similar to a "black box." It means that deep learning models lack transparency and interpretability. This is a significant drawback in many applications, where the rationale of models' decisions is a requirement for trust. Although we have built algorithms with extremely high accuracy, we can only get model parameters with unclear meaning in the end. In other words, the deep model itself contains knowledge, but humans cannot understand it. We want to know (in our way) what knowledge the model has learned from the data to make the final decision. Hence, the interpretability of deep learning is of great significance to artificial intelligence. On the one hand, it is an essential means to evaluate the safety of artificial intelligence. On the other hand, it is also conducive to accelerating the promotion of artificial intelligence applications.

Zhou et al. proposed CAM [30], one of the most representative interpretability approaches. CAM is essentially a heat map that depicts the attention information of deep learning models. They found that the weights of the classification layer, i.e., the fully connected (FC) layer after the global average pooling (GAP) layer, were highly correlative to the corresponding categories. Therefore, they propose to use the information contained in the GAP-based structure to derive CAM. In their definition, CAM is the linear weighted sum of the activation maps. For example, consider the structure that an FC layer follows a GAP layer. Let A_k denote the k-th channel of activations inputted to the GAP layer. W^c denotes the weight vector of the last FC layer with respect to class c, and its k-th element is represented by $W^c[k]$. The CAM of class c is defined as

$$L^c_{\mathrm{CAM}} = \sum_k \alpha^c_k A_k, \tag{1}$$

where

$$\alpha^c_k = W^c\,[k]. \tag{2}$$

Based on the above definition, the calculation of CAM depends on the specific structure of the FC and GAP layers. Therefore, a deep model without a GAP layer needs to be modified and retrained. Moreover, the last convolution layer is generally of small size. The CAM must always be resized to the same shape as the input image, leading to coarse spatial information after interpolating.

Grad-CAM generalizes CAM to other models without GAP layers. The core idea of Grad-CAM is to represent the fusion weights, α, by gradients. Since the calculation of gradient is independent to GAP layers, Grad-CAM is applicable in any layer. Consider a convolution layer l and a class of interest c. The prediction probability of class c is denoted as Y^c. Let A^l denote the activations of layer l, while A^l_k is the k -th channel. The spatial shape of A^l_k is $w^l \times h^l$, where w^l and h^l are, respectively, the width and height of the l-th layer in the model. The Grad-CAM, denoted as $L^c_{\mathrm{Grad-CAM}}$, is defined as

$$L^c_{\mathrm{Grad-CAM}} = \mathrm{ReLU}\left(\sum_k \alpha^c_k A^l_k\right), \tag{3}$$

where

$$\alpha^c_k = \frac{\partial Y^c}{\partial A^l_k.}. \tag{4}$$

The fusion weights α^c_k are defined by the element-wise partial derivatives of Y^c with respect to A^l_k. ReLU is adopted to remove the negative values. Grad-CAM is applicable not only for classification problems but also for other models in which the activation function can be derived.

2.2. Adversarial Example Detection. The goal of adversarial example detection is to judge if an input image is malicious. On the basis of whether to modify the input examples, existing works can be divided into two categories: (1) statistics-based and (2) denoiser-based.

2.2.1. Statistics-Based Approaches. Adversarial examples are aimed at distorting the output of their target model. Since a model commits decision divergence with high probability when facing a malicious example, there must be a statistical difference, in the example itself or the process of decision-making, between adversarial examples and the corresponding benign ones. The main idea of statistics-based approaches is to design measurable metrics for the statistical differences between adversarial and benign examples and make them as significant as possible. However, this kind of method needs some prior information, more or less. Nicolae et al. find that the adversarial examples have more significant reconstruction errors compared to the clean ones [31]. They take advantage of CapsNet [32] to reconstruct the input image and train it with L_2 loss between the input and the reconstructed image. After training, the reconstruction errors of most adversarial examples are more significant than a threshold. This method works on MNIST, Fashion MNIST, and SVHN. However, for the examples with a low distortion level, its result is not satisfactory. NIC [16] treats the detection task as a one-class classification (OCC) problem. It utilizes a one-class support vector machine (OSVM) model to classify the input images. Additional classification layers connecting to the internal layers of the original model are trained first to extract extra features, with which the OSVM can be learned. This method only needs benign examples for training, requiring no information about attack algorithms.

2.2.2. Denoiser-Based Approaches. The basic idea of denoiser-based approaches is to filter out the possible adversarial noise in the image, without destroying the original semantic information. MagNet [18] uses a reconstruction network to detect adversarial examples, which is similar to [33]. The difference is that the reconstruction network is a combination of an encoder and a decoder. After training the reconstruction network, the reconstructed image and the original image are simultaneously fed into the target model. Then, the Jensen-Shannon divergence (JSD) between the prediction logits of the two images is calculated. If the JSD goes beyond a certain threshold, the input image is considered an adversarial example. The experimental results of MagNet performed well on small sample size datasets, such as MNIST and CIFAR-10. However, Russakovsky et al. found that MagNet failed on ImageNet [34]. They propose a high-level representation guided denoiser (HGD) [19] for large images and achieve state-of-the-art results on ImageNet. Xu et al. propose the state-of-the-art denoiser feature squeezing [17]. The authors consider that the oversized input feature space is redundant for image classification. They propose to squeeze the feature space to reduce unnecessary information. Three methods are employed for denoising: squeezing color bits, median smoothing, and nonlocal smoothing. We believe that the essence of feature squeezing is to disrupt the original pixel distribution with minimal destruction of the original semantic information. However, the performance depends on artificially designed filters. Ye et al. propose a detection framework [28] based on Grad-CAM [22]. In [28], the Grad-CAM of the input image is superimposed onto the input image itself with a particular ratio to generate an emphasized image I^E:

$$I^E = I + \theta * L^c_{\text{Grad-CAM}}. \tag{5}$$

I represents the input image and θ is a hyperparameter. $L^c_{\text{Grad-CAM}}$ is calculated by formula (3).

Then, the original input image and the emphasized image are simultaneously fed into the same deep model to compare their prediction labels. If the prediction results are not the same, the original input image is considered malicious.

3. Adversarial Example Detection

In this section, we first explore the problem of deep learning models from the perspective of adversarial attack and defense. Based on the discussion, the design philosophy of our work is put forward. Then, the reasons why Score-CAM is chosen are discussed. At last, the algorithm framework and its running procedure are described.

3.1. Design Philosophy

3.1.1. Denoising Motivation. Noise in a specific range in images usually has no harm to the performance of DNNs. There are already mature skills to enhance the robustness of DNN models, including data augmentation, transfer learning, and dropout. A DNN model can be trained to work well in various scenarios with different noise levels.

However, the situation is different for adversarial examples. One of the primary principles of attack methodology is to impact the final output as much as possible by using the slightest change to the input. The level of adversarial distortion will accumulate along with the depth, which has been proved by some previous studies [1, 17, 24, 26]. Slight as the malicious perturbation is, adversarial examples are sensitive to even low-level random noises.

Based on the above discussion, an instinctive idea is to cover the perturbations with random noise. However, superimposing the whole image by random noise with no difference in intensity may cause unnecessary information loss. Therefore, adding appropriate noise with the slightest affection to the benign examples' accuracy becomes the key to the problem. CAM provides a superior solution for this problem.

CAM is designed for deep learning interpretability, making it a suitable tool to reflect the inside activation state. It reveals the internal information of deep models by visualization method. Given an input image and a class of interest, CAM draws the heat map that indicates the contribution of each area (in the input image) to the prediction score. In other words, it reveals the spatial activation level of a chosen layer. For an unsoiled picture, CAM correctly displays the activation state w.r.t. the ground truth label. For a malignant image that tutors the target model to make a wrong classification decision, the deliberate alteration will change the neurons' activation mode in the target model. Based on the mistakenly predicted class, CAM will capture the abnormal activation of neurons and express it through the heat map. Figure 1 shows the juxtapositions of CAMs from some benign examples and their corresponding adversarial examples. The visualization results before and after being polluted by adversarial perturbations are displayed in three groups: input images (Figures 1(a) and 1(b)), Grad-CAM (Figures 1(c) and 1(d)), and Score-CAM (Figures 1(e) and 1(f)). It demonstrates that from the perspective of no matter Grad-CAM or Score-CAM, the model's interesting areas are manipulated by the unperceivable modifications in the input. For example, Figures 1(c) and 1(e) show that the attention of the model is on the area of the main objects (a boy in a go-kart) when the input is original images. But adversarial noises switch the hot zone to the background (Figures 1(d) and 1(f)). Hence, we can exploit the difference of the intermediate information between the unstained and the antagonistic examples to trim the random noise imposed on the detected examples.

We propose to denoise the adversarial perturbations by superimposing the input image with random noise weighted by CAM in the spatial dimension. More specifically, a random Gaussian noise matrix of the same size as the input example is first generated. Afterward, the noise is dot product with the CAM. Finally, the input image (not sure whether clean) is covered by the noise edited by CAM. Consequently, the region with a higher activation level is embedded with higher-level noise after the above transformation. For benign examples, noise covered in the interested area, where the most potential features are located, may lead to a partial loss of information. Nevertheless, the primary semantic information cannot be wrecked if the noise level is controlled to a certain level. The model can still take advantage of the information in the denoised image to make decisions. In contrast, if the input is a poisoned example, the predicted class differs from the ground truth label. CAM will draw the heat map based on the wrong class, where the activated area is different from the area with the wealthiest semantic information. Hence, the edited noise trimmed by the heat map may slightly affect the original area with semantic information. But the distribution of the adversarial perturbations could be distorted more severely. Meanwhile, note that they are deliberately designed to be as small as possible.

The first line in Figure 1 can be a more intuitive example to explain our motivation. As depicted in Figure 1(e), Score-CAM accurately sketches the bird's contour that contains the wealthiest semantic information. If the input is a clean example (Figure 1(a)), the random noise will cover the bird's area in line with the result (Figure 1(e)). The decorated noise only hinders the classification slightly based on the previous

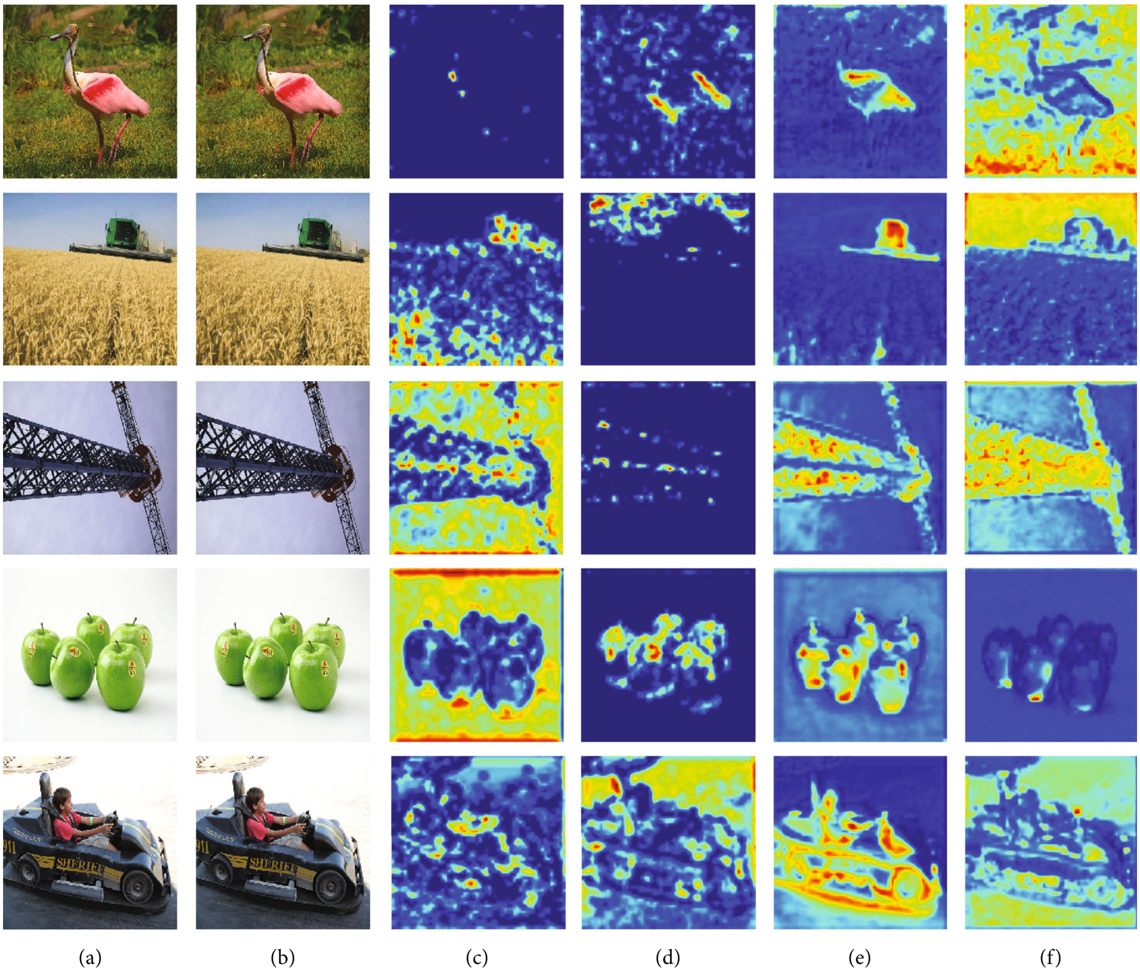

Figure 1: Visualization results: (a) original image; (b) adversarial example; (c) original Grad-CAM; (d) adversarial Grad-CAM; (e) original Score-CAM; (f) adversarial Score-CAM.

discussion. In contrast, as for the adversarial example (Figure 1(b)), the attack algorithm switches the high-light zone to the pixels of grassland (shown in Figure 1(f)). When this contaminated example is fed into our framework, the background region is full of emphasized Gaussian noise. Nevertheless, the principal entity, a bird, is barely influenced because we trim the noise's value according to the Score-CAM.

Section 3.2.1 will hand out a more detailed description of the proposed detection framework.

3.1.2. Choice of CAM. Most methods for extracting CAM are based on gradient. However, gradient-based methods have flawed characteristics and disadvantages to reveal the real attention of DNN models. First, for a DNN model with dozens of layers, gradient vanishing caused by activation functions cannot be ignored. For example, there is the inconsistency of gradient caused by the flat zero-gradient interval in the ReLU function, one of the most used activation functions. The inconsistency could bring about high-frequency spatial noises while computing the output gradient for an internal activation map. Second, the gradient is likely to conduct false confidence due to gradient saturation. The area highlighted by the gradient does not always contribute proportional confidence to the result. This phenomenon is discovered by [26]. Last but not least, most real-world deployment environments, e.g., edge computing environments [35], cannot support the gradient computation of deep models. Moreover, neural network quantization is also widely utilized for deep model deployment, resulting in higher complexity and more significant error in computing gradient. The above facts mean that gradient-based techniques, like Grad-CAM, are not universally applicable.

Score-CAM [26] adopts gradient-free method to design the fusion weights, i.e., α. It introduces the concept of channel-wise increase of confidence (CIC) to measure the importance of the activation map in each channel. It utilizes the image masked by the activation in each channel to compute CIC. The linear sum of activations weighted by CIC is further calculated. Given a DNN model and a class of interest c, the function $Y^c = f(X)$ takes an image X and outputs a scalar Y^c that represents the output probability for class c. Let A^l denote the activation of the l-th convolutional layer, and A_k^l denotes the k-th channel of A^l. The Score-CAM of class c is formulated by two steps:

(1) Computing CIC

Considering a known baseline input X_b, the contribution of A_k^l towards Y^c is defined as

$$C\left(A_k^l\right) = f\left(X \cdot H_k^l\right) - f(X_b), \tag{6}$$

where

$$H_k^l = \text{norm}\left(\text{Up}\left(A_k^l\right)\right). \tag{7}$$

In this paper, X_b is a zero matrix with the same size of X. $\text{Up}(A_k^l)$ denotes upsampling A_k^l to the same spatial size as original input X. norm $(\bullet)$ is a min-max normalization function that limits the raw activation values in $[0, 1]$.

(2) Computing Score-CAM

In the process of calculating Score-CAM, $C(A_k^l)$ is the weighted mask for k-th channel. By applying the weighted masks to the original activation maps, we can get the Score-CAM:

$$L^c_{\text{Score-CAM}} = \text{ReLU}\left(\sum_k C\left(A_k^l\right) \cdot A_k^l\right). \tag{8}$$

The ReLU function is used for the disturbance of irrelevant pixels on the activation map.

In the experiments conducted by [26], Score-CAM performs better than Grad-CAM [22] and Grad-CAM++ [23] in no matter the visualization of the heat map or the quantitative evaluation. The visualized results of these two CAM methods are depicted in Figure 1. Figures 1(c) and 1(e), respectively, show the results of Grad-CAM and Score-CAM. Score-CAM can always highlight the main objects and suppress the noise in the background area, while Grad-CAM obtains the inaccurately activated heat map on most occasions. Combining the above analysis, we believe that Score-CAM can play a better role than the methods based on the gradient in the adversarial defense.

3.2. Detection Framework Design

3.2.1. Detecting the Adversarial Examples. At present, we have obtained the activated map containing the attention information of the model. The next question is how to use this information to distinguish out the vicious examples.

Our approach is to denoise the adversarial perturbations with decorated noise. We depict the detection framework in Figure 2.

The computation process of the denoised image I^* from an original image I can be formulated as:

$$I^* = I + N \cdot L^c_{\text{Score-CAM}}. \tag{9}$$

First, we generate a noise matrix N with the same shape as I. Then, we compute the weighted noise by dot-multiplying the noise matrix N and the Score-CAM of I w.r.t. the class of interest c, i.e., $L^c_{\text{Score-CAM}}$. In this paper, we adopt the class with the highest predicted probability as the class of interest, and the Score-CAM is default resized to the same shape as the input image and the noise matrix. Last, the weighted noise and original image are added to generate a new image I^* called edited image. Here, we directly trim the pixel values beyond [0,255]. We utilize random Gaussian noises with zero mean value and an adjustable standard deviation σ.

This method introduces randomization to the defense side to lower the possibility of being bypassed by targeted malicious attacks. Besides, it does not shift the mean value of the original pixels' distribution and does not severely degrade the prediction accuracy of the models.

The last part of the detection framework is the mechanism of result determination. Based on the discussion in Section 3.1, noise with a limited level only weakly affects the recognition of the benign example. On the contrary, weak noises can lead to the failure of adversarial perturbations since they are designed to be as unperceivable and tiny as possible. The prediction results of the original image I and the edited image I^* will be compared to judge if I is adversarial. If the two images correspond to different prediction labels, the original image I is determined an adversarial example. On the contrary, consistent prediction of the two images hints at a clean example.

3.2.2. Setup for Score-CAM. Different attack algorithms behave differently in altering input pictures and manipulating cells. Some adversarial algorithms such as L_{inf} attacks limit the magnitude of changed pixels rather than pixel numbers. Malicious examples tend to activate large numbers of neurons abnormal to the actual labels. By accumulating a considerable amount of tiny deviations, qualitative change happens and the prediction label changes. On the contrary, L_0 attacks limit the number of pixels modified. They tend to exploit a few amplifier paths and lead to a decisive change in deeper layers. Most attacks exploit both aforementioned ways, such as L_2 attack, which constrains the total change using the Euclidean distance to produce more unperceivable perturbations. For both L_{inf} and L_0 attacks, the shallow layers in the target model often do not accumulate significant adversarial disturbances. The drastic changes may occur in a deeper layer. Therefore, shallow layers are not the ideal targets for extracting CAM in our work.

In the process of calculating Score-CAM, activation maps are upsampled to the same spatial size as the input image. After that, the resized activation maps will be used as the mask onto the input image. However, it is a "first-line therapy" to reduce the spatial size along the inference direction when designing a convolution network. For

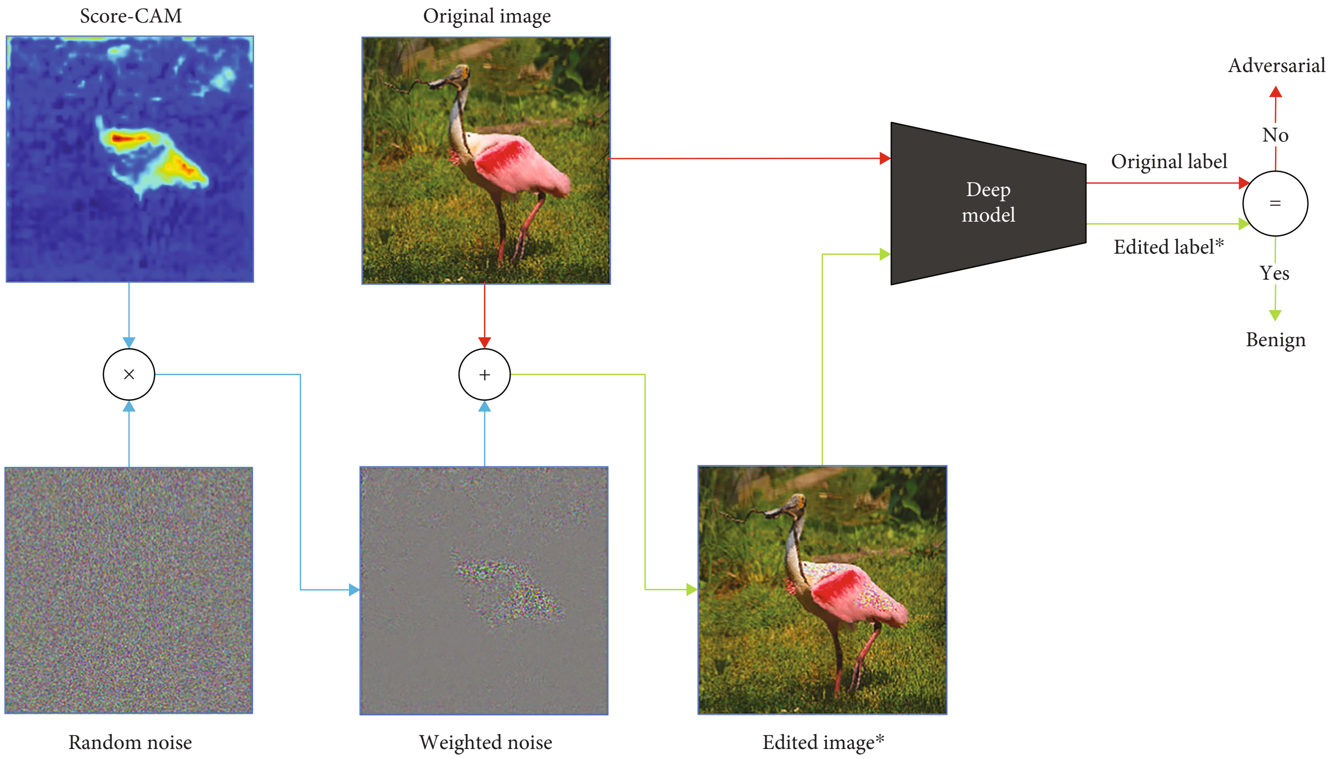

Figure 2: Detection framework based on Score-CAM-decorated noise.

Table 1: Details of target models.

Models	Attack success rate (%)						Model accuracy (%)	Layer name for CAM	Activation shape
	BIM (L_{inf})	PGD (L_2)	PGD (L_1)	FGSM (L_{inf})	CW (L_{inf})	CW (L_2)			
ResNet50	93.34	92.28	91.07	93.19	90.32	92.05	68.08	conv3_block4_out	28*28*512
ResNet101	92.05	89.76	88.70	93.19	91.42	88.70	69.98	conv3_block4_out	28*28*512
DenseNet201	94.15	92.44	96.01	92.30	92.15	91.97	74.49	pool3_relu	28*28*512
Xception	91.83	90.76	90.63	89.75	94.78	92.50	77.52	block4_sepconv2_act	37*37*728
InceptionV3	92.68	94.26	91.05	90.34	95.72	88.90	76.28	mixed2	35*35*288

example, the size of activation maps in the last convolution layer of ResNet50 is 7*7 while inputting an image with the size of 224*224. According to formula (8), the output size of Score-CAM is dependent on the spatial size of the l-th layer's activation map. So, the size of Score-CAM is usually smaller than the input image. However, the Score-CAM will be resized to the shape of the input image according to formula (9) by using the interpolation algorithm (nearest-neighbor interpolation in our implementation). Therefore, the spatial information is too coarse for extracting Score-CAM if we use the activation map from the very deep layers.

After the above discussion, we can conclude that the layers in the middle of a model are most appropriate for our framework. The specific layer names and the size of the activation maps are listed in Table 1. We also conduct an ablation experiment to verify our inference in Section 4.3.

4. Evaluation

In this section, we conduct experiments to evaluate the effectiveness of the proposed detection framework.

4.1. Implementation Details

4.1.1. Dataset and Models. We conduct experiments on ILSVRC2012 samples from ImageNet [34], one of the most representative colored image datasets for computer vision tasks. Several prevalent DNNs are chosen as the target models, including ResNet50, ResNet101, DenseNet201, Xception, and InceptionV3. They are recently the most prevailing architectures and are used as backbone networks in all kinds of computer vision tasks, such as face recognition, semantic segmentation, and object detection. The pretrained model weights and preprocessing API come from Keras.

Table 2: Results for adversarial example detection.

Attack	Method	CAM	ResNet50			ResNet101			DenseNet201			Xception			InceptionV3		
			Hyperparameter	Adversaries' accuracy	Success rate	Hyperparameter	Adversaries' accuracy	Success rate	Hyperparameter	Adversaries' accuracy	Success rate	Hyperparameter	Adversaries' accuracy	Success rate	Hyperparameter	Adversaries' accuracy	Success rate
BIM (L_{inf})	WN	Grad-CAM	34.41	23.60%	34.00%	40.91	30.52%	39.36%	42.46	31.20%	39.72%	56.53	21.56%	31.88%	76.95	36.80%	44.36%
		Score-CAM	34.72	48.00%	59.24%	31.86	53.28%	62.48%	26.68	59.16%	67.28%	30.08	52.84%	60.96%	63.49	59.88%	66.68%
	EI	Grad-CAM	0.33	16.40%	25.80%	0.31	19.00%	27.00%	0.47	28.00%	38.60%	0.55	16.00%	32.40%	0.62	29.40%	39.00%
		Score-CAM	0.38	18.40%	26.40%	0.5	29.80%	39.20%	0.49	34.20%	44.20%	0.52	28.20%	42.40%	0.88	32.00%	39.20%
PGD (L_2)	WN	Grad-CAM	33.02	24.32%	32.08%	37.72	27.82%	30.71%	40.6	27.40%	33.72%	56.38	22.92%	32.44%	78.96	32.96%	40.00%
		Score-CAM	34.72	52.00%	60.52%	30.83	50.80%	58.97%	25.85	57.28%	65.52%	30.99	59.76%	66.80%	66.59	56.84%	65.08%
	EI	Grad-CAM	0.31	15.80%	25.80%	0.31	21.15%	26.28%	0.45	26.20%	34.80%	0.55	20.60%	33.40%	0.62	24.40%	35.40%
		Score-CAM	0.34	18.60%	25.00%	0.38	20.51%	25.00%	0.49	32.60%	41.20%	0.52	32.00%	42.20%	1	30.80%	40.60%
PGD (L_1)	WN	Grad-CAM	30	27.76%	35.64%	41.22	30.84%	39.24%	41.53	23.20%	29.80%	55.75	24.24%	32.60%	76.49	37.56%	45.00%
		Score-CAM	31.78	52.28%	61.88%	33.03	49.12%	58.44%	26.83	53.80%	61.96%	31.09	56.28%	63.68%	62.57	61.28%	68.56%
	EI	Grad-CAM	0.31	21.20%	31.40%	0.31	17.20%	23.00%	0.47	24.40%	34.20%	0.55	19.60%	34.20%	0.59	28.00%	37.00%
		Score-CAM	0.34	23.20%	30.40%	0.56	25.80%	35.00%	0.49	28.80%	36.40%	0.52	30.00%	40.00%	0.81	29.20%	36.00%
FGSM (L_{inf})	WN	Grad-CAM	33.02	9.56%	29.64%	42.4	8.12%	31.56%	40.29	15.40%	31.36%	32.56	16.35%	33.52%	60.20	13.87%	33.56%
		Score-CAM	32.94	18.76%	38.04%	31.94	13.64%	31.00%	26.29	28.64%	46.44%	22.45	22.58%	43.26%	46.54	15.68%	38.54%
	EI	Grad-CAM	0.33	4.80%	29.40%	0.31	3.20%	27.40%	0.45	12.00%	34.20%	0.41	15.34%	32.68%	0.48	9.78%	33.21%
		Score-CAM	0.38	5.00%	23.00%	0.56	5.40%	30.00%	0.49	17.20%	36.60%	0.26	10.56%	28.96%	0.65	9.21%	29.54%
CW (L_{inf})	WN	Grad-CAM	31.78	49.72%	61.04%	40.29	42.48%	56.28%	41.68	54.16%	62.68%	58.54	69.56%	76.36%	76.56	64.64%	74.24%
		Score-CAM	32.25	60.56%	69.84%	31.94	52.72%	65.60%	25.98	70.88%	78.96%	32.4	82.36%	87.36%	68.75	79.64%	85.64%
	EI	Grad-CAM	0.31	45.00%	57.00%	0.31	30.40%	44.20%	0.45	54.20%	65.40%	0.55	68.40%	77.80%	0.62	62.60%	73.80%
		Score-CAM	0.34	39.20%	53.20%	0.5	33.00%	50.40%	0.49	53.40%	65.40%	0.53	66.40%	77.00%	1.00	67.40%	78.00%
CW (L_2)	WN	Grad-CAM	31.01	51.60%	64.28%	40.14	51.16%	65.24%	41.38	69.60%	76.56%	56.53	53.16%	63.36%	79.71	31.65%	51.39%
		Score-CAM	33.17	61.84%	72.56%	32.56	57.32%	70.68%	25.75	78.32%	85.32%	31.67	63.08%	72.84%	74.61	41.77%	57.47%
	EI	Grad-CAM	0.31	46.00%	59.40%	0.31	42.40%	57.40%	0.45	65.20%	75.80%	0.55	49.20%	66.60%	0.69	32.91%	59.49%
		Score-CAM	0.34	37.40%	50.60%	0.5	37.00%	53.40%	0.49	59.80%	71.20%	0.53	46.40%	64.00%	1.04	29.11%	56.96%

Table 3: Defense performance comparison between layers.

Layer name	Activation shape	BIM (L_{inf})			PGD (L_2)		
		Hyperparameter (σ)	Adversaries' accuracy	Success rate	Hyperparameter (σ)	Adversaries' accuracy	Success rate
conv1_relu	112*112*64	33.28	47.97%	57.40%	25.82	45.80%	53.64%
conv2_block3_out	56*56*256	31.54	51.38%	59.67%	27.89	47.54%	55.28%
conv3_block4_out	28*28*512	31.86	53.28%	62.48%	30.83	50.80%	58.97%
conv4_block23_out	14*14*1024	23.67	50.24%	56.59%	22.24	50.38%	57.24%
conv5_block3_out	7*7*2048	26.14	37.97%	44.63%	19.95	38.06%	43.62%

Table 4: Comparison of different OSPA.

BIM (L_{inf})				PGD (L_2)			
Hyperparameter (σ)	OSPA	Adversaries' accuracy	Success rate	Hyperparameter (σ)	OSPA	Adversaries' accuracy	Success rate
113.83	59.75%	51.60%	86.01%	113.73	60.00%	51.54%	85.38%
88.92	70.06%	57.18%	82.76%	83.96	70.06%	57.12%	82.18%
54.32	80.31%	61.96%	76.38%	51.42	80.13%	60.32%	73.46%
31.86	90.04%	53.28%	62.48%	30.83	89.94%	50.38%	58.97%
18.66	95.09%	36.69%	42.70%	19.00	94.87%	37.50%	42.12%

Eight-bit images are converted to float matrixes. Afterward, the alterations in our experiments are directly conducted on these matrixes with restricted values from 0 to 255.

4.1.2. Attack Setup. The adversarial examples are generated based on the images which are correctly classified by the target models from ILSVRC2012-val. For each target model and each attack algorithm, we select 500 successful adversarial examples and the corresponding original images as the test data.

In other words, every detection experiment is conducted on a test set containing 500 benign examples and 500 corresponding adversarial examples.

Our experiments are conducted with several representative white-box adversarial attack algorithms: FGSM [3], BIM [36], PGD [10], and CW attacks [6]. Attacks with different norms are also taken into consideration. In this paper, we adopt untargeted attacks for our experiments.

Low-level disturbance for adversarial examples is one of the development targets. To avoid generating coarse adversarial image examples, we tune the hyperparameters of attacks carefully and keep the attack success rate around 90%. We adopt the implementations from the ART library [31]. The details of the attacks and target models are listed in Table 1.

4.2. Adversarial Example Detection. Table 2 shows the experimental results on adversarial example detection of the proposed method (weighted noise (WN) with Score-CAM, written as Score-CAM+WN for abbreviation) versus the state-of-the-art method (emphasized image (EI) with Grad-CAM, written as Grad-CAM+EI for abbreviation) proposed by Ye et al. [28]. For the completeness of the experiment, we also introduce two other ablation experiments, i.e., Score-CAM+EI and Grad-CAM+WN. The hyperparameter σ denotes the standard deviation of Gaussian white noise employed only in WN, and θ is the proportion of CAM emphasized to an image used only in EI.

For all the above methods, editing the input examples disturbs the original pixel distributions, leading to accuracy degradation on the original benign examples. This accuracy is called Original Samples Prediction Accuracy (OSPA). In our experiments, OSPA is 100% when the hyperparameter σ or θ equals zero. It is because the chosen examples are not edited at this time, and all of them can be correctly classified. OSPA will decrease along with the increase of σ or θ. To fairly compare the effectiveness of different approaches, OSPA is adjusted to 90% (±0.5%) for different experiments by tuning σ or θ of the corresponding method.

The experiments consist of 30 groups: 6 attacks * 5 models. The left-most column shows the attack name and its norm type. For example, CW (L_2) indicates the Carlini and Wagner attack with L_2 norm. The top row shows the names of the six target models. We demonstrate three values for each experiment: hyperparameter, adversarial example accuracy (adversaries' accuracy), and detection success rate (success rate). The detection success rate is the percentage of the examples with different prediction labels before and after being edited in 500 adversarial examples. Adversarial example accuracy is the prediction accuracy of adversarial examples after being denoised.

Since random noises are introduced into the detection framework, the results are not the same for each time. Therefore, 10-fold testing is applied in the WN method. For each experiment introducing random noises, the final result is the average value of 10 times repeat.

As shown in Table 2, except for the FGSM attack, the detection success rate of the proposed method reaches more than 60% in most cases. When facing FGSM attacks, there is a drop in the success rate. We believe that the FGSM attack is relatively coarse. So greater distortion level (greater step size) is needed to maintain the attack success rate of 90%.

To maintain the attack success rate of 90% in our experiments, greater step size is adopted and higher-level distortion is added. The decorated weighted noises with the same σ or θ could not decompose the adversarial perturbations.

The very noticeable point is that the proposed method (Score-CAM+WN) achieves a higher success rate than the baseline (Grad-CAM+EI) in almost all the cases. Even in the experiments where the proposed method has poorer performance (CW (L_2) and Xception), its gap to the best is insignificant. It proves that the proposed method is more sensitive to adversarial examples. Score-CAM always performs better by comparing the results of the same CAM type but the different superimposing methods. By comparing the results of the same superimposing method but different CAM types, WN always performs better. The data of adversarial examples accuracy shows a similar pattern.

Considering that no training is carried out before deployment, the proposed method achieves quite impressive results. Furthermore, it works for different attacks and various models, demonstrating its generality.

4.3. Choice of Layer. In this section, we validate the analysis and discussion about different layers in Section 3.2.2. Activations from different layers are utilized to generate Score-CAM. Furtherly, the images are edited by WN. ResNet101 is chosen as the target model. The adversarial examples are produced by BIM (L_{inf}) and PGD (L_2), as described in Table 1. Five layers are picked out for this evaluation. Each layer is the output of the last one in the bottleneck blocks with the same shape. For example, there are four bottleneck blocks with an output shape of 28*28*512: conv3_block1 to conv3_block4, and conv3_block4 is the last one.

As shown in Table 3, conv3_block4_out with the output shape of 28*28*512 performs best. The defense results rise first and then descend along with the reduction of the spatial size. It is fully in line with our previous analysis in Section 3.2.2.

Another noticeable phenomenon is that σ descends with shrinking spatial size, in general. Since we keep the OSPA at 90% for all experiments, this phenomenon indicates that a lower noise level is needed to maintain OSPA when using Score-CAM with a smaller spatial size.

4.4. The Trade-Off between OSPA and Success Rate. In this section, we provide a survey of the relationship between OSPA and detection success rate. The adversarial examples are still produced on ResNet101 by BIM (L_{inf}) and PGD (L_2). The attack configuration is the same as described in Table 1. As shown in Table 4, our method reaches more than a 42% success rate at the OSPA of 90% for both BIM and PGD attacks. The success rate improves along with the descend of OSPA. However, the accuracy of adversarial examples first increases and then decreases. Decorated noise added to contaminated images can mitigate the adverse effects of adversarial perturbations. Hence, the adversaries' accuracy increases first. However, DNN can only filter out random noise within a certain limit. When the noise power is too large, the original semantic information will be wrecked. This leads to a drop in the adversaries' accuracy.

5. Conclusion

In this paper, we propose a gradient-independent adversarial example detection framework based on the technique of deep learning interpretability. Based on the discussion, we conclude that adversarial examples are sensitive to random noise while clean ones are not. We cover the perturbations with decorated random noise by taking advantage of this property. The random noise is decorated based on the example-wise Score-CAM to emphasize the area where the target model really focused and to eliminate unnecessary accuracy loss. Extensive experimental results show that the proposed framework can always achieve the highest prediction accuracy and detection success rate compared with previous works. We further make ablation experiments to explore the impact of Score-CAM from different layers and find that the middle layer of models is most suitable to extract Score-CAM. In addition, we also investigate the trade-off between clean data accuracy and detection success rate. We believe that our framework can be easily updated when more accurate and efficient saliency map methods emerge.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62173066).

References

[1] X. Chen and G. Jin, "Preschool education interactive system based on smart sensor image recognition," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 2556808, 11 pages, 2022.

[2] K. Zhang, H. Ying, H. N. Dai et al., "Compacting deep neural networks for Internet of Things: methods and applications," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 11935–11959, 2021.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," pp. 1–11, 2015, https://arxiv.org/abs/1412.6572.

[4] H. Kwon and S. Kim, "Restricted-area adversarial example attack for image captioning model," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 9962972, 9 pages, 2022.

[5] X. Fu, Z. Gu, W. Han, Y. Qian, and B. Wang, "Exploring security vulnerabilities of deep learning models by adversarial attacks," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 9969867, 9 pages, 2021.

[6] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 39–57, San Jose, CA, USA, 2016.

[7] X. Wei, Y. Guo, and B. Li, “Black-box adversarial attacks by manipulating image attributes,” *Information Sciences*, vol. 550, pp. 285–296, 2021.

[8] T. Yang, X. Zhao, X. Wang, and H. Lv, “Evaluating facial recognition web services with adversarial and synthetic samples,” *Neurocomputing*, vol. 406, pp. 378–385, 2020.

[9] S. Qiu, Q. Liu, S. Zhou, and W. Huang, “Adversarial attack and defense technologies in natural language processing: a survey,” *Neurocomputing*, vol. 492, pp. 278–307, 2022.

[10] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” pp. 1–28, 2018, https://arxiv.org/abs/1706.06083.

[11] K. Ren, T. Zheng, Z. Qin, and X. Liu, “Adversarial attacks and defenses in deep learning,” *Engineering*, vol. 6, no. 3, pp. 346–360, 2020.

[12] C. Zhu, Y. Cheng, Z. Gan, S. Sun, T. Goldstein, and J. Liu, “FreeLB: enhanced adversarial training for natural language understanding,” pp. 1–12, 2020, https://arxiv.org/abs/1909.11764.

[13] N. Carlini and D. Wagner, “Adversarial examples are not easily detected: bypassing ten detection methods,” in *Proceedings of the ACM Workshop on Artificial Intelligence and Security*, pp. 3–14, New York, NY, USA, 2017.

[14] X. Li, X. Zhang, F. Yin, and C. L. Liu, “Decision-based adversarial attack with frequency mixup,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1038–1052, 2022.

[15] G. Tao, S. Ma, Y. Liu, and X. Zhang, “Attacks meet interpretability: attribute-steered detection of adversarial samples,” in *Advances in Neural Information Processing Systems*, pp. 7717–7728, Palais des Congrès de Montréal, 2018.

[16] S. Ma, Y. Liu, G. Tao, W.-C. Lee, and X. Zhang, “NIC: detecting adversarial samples with neural network invariant checking,” in *Proceedings of the Network and Distributed System Security Symposium*, pp. 1–15, Indiana USA, 2019.

[17] W. Xu, D. Evans, and Y. Qi, “Feature squeezing: detecting adversarial examples in deep neural networks,” in *Proceedings of the Network and Distributed System Security Symposium*, pp. 1–15, Virginia, USA, 2018.

[18] D. Meng and H. Chen, “Magnet: a two-pronged defense against adversarial examples,” in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147, New York, NY, USA, 2017.

[19] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, “Defense against adversarial attacks using high-level representation guided denoiser,” in *Proceedings of the IEEE on Computer Vision and Pattern Recognition*, pp. 1778–1787, Salt Lake City, UT, USA, 2018.

[20] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser et al., “Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI,” *Information Fusion*, vol. 58, pp. 82–115, 2020.

[21] Y. Liang, S. Li, C. Yan, M. Li, and C. Jiang, “Explaining the black-box model: a survey of local interpretation methods for deep neural networks,” *Neurocomputing*, vol. 419, pp. 168–182, 2021.

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626, Venice, Italy, 2017.

[23] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, “Grad-CAM++: generalized gradient-based visual explanations for deep convolutional networks,” in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 839–847, Lake Tahoe, NV, USA, 2018.

[24] D. Omeiza, S. Speakman, C. Cintas, and K. Weldermariam, “Smooth Grad-CAM++: An Enhanced Inference Level Visualization Technique for Deep Convolutional Neural Network Models,” 2019, https://arxiv.org/abs/1908.01224.

[25] S. Desai and H. G. Ramaswamy, “Ablation-CAM: visual explanations for deep convolutional network via gradient-free localization,” in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 972–980, Bordeaux, France, 2020.

[26] H. Wang, Z. Wang, M. Du et al., “Score-CAM: score-weighted visual explanations for convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 111–119, Seattle, WA, USA, 2020.

[27] S. Wang and Y. Gong, “Adversarial example detection based on saliency map features,” *Applied Intelligence*, vol. 52, no. 6, pp. 6262–6275, 2021.

[28] D. Ye, C. Chen, C. Liu, H. Wang, and S. Jiang, “Detection defense against adversarial attacks with saliency map,” *International Journal of Intelligent Systems*, vol. 37, pp. 1–18, 2021.

[29] Z. Zhao, H. Duan, G. Min et al., “A lighten CNN-LSTM model for speaker verification on embedded devices,” *Future Generation Computer Systems*, vol. 100, pp. 751–758, 2019.

[30] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, Las Vegas, NV, USA, 2016.

[31] M. Nicolae, M. Sinn, M. N. Tran et al., “Adversarial Robustness Toolbox v1.0.0,” 2018, https://arxiv.org/abs/1807.01069.

[32] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3857–3867, Toronto, 2017.

[33] N. Frosst, S. Sabour, and G. Hinton, *Darccc: Detecting Adversaries by Reconstruction from Class Conditional Capsules*, 2018, https://arxiv.org/abs/1811.06969.

[34] O. Russakovsky, J. Deng, H. Su et al., “ImageNet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[35] W. Zhan, C. Luo, J. Wang et al., “Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing,” *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5449–5465, 2020.

[36] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Artificial Intelligence Safety And Security*, pp. 99–112, Chapman and Hall/CRC, 2018.