Fall 2021

Towards More Trustworthy Deep Learning: Accurate, Resilient, and Explainable Countermeasures Against Adversarial Examples

Fei Zuo

Towards More Trustworthy Deep Learning: Accurate, Resilient, and Explainable Countermeasures Against Adversarial Examples

by

Fei Zuo

Master of Economics
Sun Yat-sen University, 2013

Master of Science
The University of Melbourne, 2016

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Science

College of Engineering and Computing

University of South Carolina

2021

Accepted by:

Qiang Zeng, Major Professor

Csilla Farkas, Committee Member

Lannan Luo, Committee Member

Song Wang, Committee Member

Xiaofeng Wang, Committee Member

Tracey L. Weldon, Interim Vice Provost and Dean of the Graduate School

Dedication

This dissertation is dedicated to my dear parents whose unwavering love and support always encourage me to pursue a greater achievement.

Acknowledgments

First of all, I would like to express my deepest gratitude to my advisor, Prof. Qiang Zeng, for his patient guidance, inspiring opinions and encouragement in accomplishing the research for this dissertation. Prof. Zeng has broadened my scope of knowledge, widened my horizon, and led me to step into the world of cybersecurity. Without the insightful comments and persistent support from Prof. Zeng, my doctorate study would never be completed.

In addition, I am sincerely grateful for the support and help from my dissertation committee members, Prof. Csilla Farkas, Prof. Lannan Luo, Prof. Song Wang and Prof. Xiaofeng Wang. They provide valuable feedback and inspiring suggestions on my dissertation, which polished this work to be better. It is a great honor for me to have them as the committee members.

Furthermore, I truly appreciate all my collaborators Lannan Luo, Xiaopeng Li, Fengyao Yan, Patrick Young, Xiaojiang Du and Chenglong Fu. In particular, I would like to extend special thanks to Prof. Lannan Luo for her precious insights and constructive advice when discussing my research.

I also would like to thank all the members of Prof. Zeng and Prof. Luo's labs. The friendly and encouraging atmosphere provided by you made me in-depth enjoy both my research and campus life during the recent years.

Last but not least, I am deeply indebted to my parents for their unlimited love and support over my years of study.

This research was supported in part by US National Science Foundation under CNS-1856380, CNS-1815144, CNS-1850278 and CNS-2016415.

Abstract

Despite the great achievements made by neural networks on tasks such as image classification, they are brittle and vulnerable to adversarial example (AE) attacks, which are crafted by adding human-imperceptible perturbations to inputs in order that a neural-network-based classifier incorrectly labels them. Along with the prevalence of deep learning techniques, the threat of AEs attracts increasingly attentions since it may lead to serious consequences in some vital applications such as disease diagnosis.

To defeat attacks based on AEs, both detection and defensive techniques attract the research community's attention. Given an input image, the detection system outputs whether it is an AE, so that the target neural network can reject those adversarial inputs. A defense technique, given an AE, helps the target neural network make correct prediction by either rectifying the AE or fortifying the classifier itself.

While many countermeasures against AEs have been proposed, recent studies show that the existing detection methods usually goes ineffective when facing adaptive AEs. In this work, we exploit AEs by identifying their noticeable characteristics.

First, we noticed that L_2 adversarial perturbations are among the most effective but difficult-to-detect attacks. How to detect adaptive L_2 AEs is still an open question. At the same time, we find that, by randomly erasing some pixels in an L_2 AE and then restoring it with an inpainting technique, the AE, before and after the steps, tends to have different classification results, while a benign sample does not show this symptom. We thus propose a novel AE detection technique, Erase-and-Restore (E&R), that exploits the intriguing sensitivity of L_2 attacks. Comprehensive experiments conducted on standard image datasets show that the proposed detector

is effective and accurate. More importantly, our approach demonstrates strong resilience to adaptive attacks. We also interpret the detection technique through both visualization and quantification.

Second, previous work considers that it is challenging to properly alleviate the effect of the heavy corruptions caused by L_0 attacks. However, we argue that the uncontrollable heavy perturbation is an inherent limitation of L_0 AEs, and thwart such attacks. We thus propose a novel AE detector by converting the detection problem into a comparison problem. More concretely, given an image I, it is pre-processed to obtain another image I'. Then, a well-trained Siamese network automatically and precisely captures the discrepancies between I and I' to detect L_0 perturbations. In addition, we show that the pre-processing technique used for detection can also work as an effective defense, which has a high probability of removing the adversarial influence of L_0 perturbations. Thus, our system demonstrates not only high AE detection accuracies, but also a notable capability to correct the classification results.

Finally, we propose a comprehensive AE detector which systematically combines the two aforementioned detection methods to thwart all categories of widely discussed AEs, i.e., L_0, L_2, and L_∞ attacks. By acquiring the both strengths from its assembly components, the new hybrid AE detector is not only able to distinguish various kinds of AEs, but also has a very low false positive rate on benign images. More significantly, through exploiting the noticeable characteristics of AEs, the proposed detector is highly resilient to adaptive attack, filling a critical gap in AE detection.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Recent years have witnessed tremendous success of neural networks in a variety of fields, such as object detection [1], motion tracking [2], and face recognition [3, 4]. Despite these great achievements, they are vulnerable to adversarial examples (AEs). Szegedy et al. [5] analyze the robustness of neural networks when facing adversarial attacks, and show that deep learning systems are sensitive to small adversarial perturbations. A neural-network-based classifier thus can be misled by AEs and generate incorrect classification results.

The term *adversarial example* can be formally defined as following. For a pre-trained neural network f, let x be an original image. An adversarial example x^{adv} is such an intentionally designed input by attackers which can guide the model f to make an incorrect prediction. Moreover, to hide the adversarial perturbation, the generation of x^{adv} is equivalent to solve the following constrained optimization problem:

$$\begin{aligned} &\min_{x^{adv}} \|x^{adv} - x\|_p \\ &\text{s.t.} \quad \bar{y} = f(x^{adv}) \\ &\qquad y = f(x) \\ &\qquad y \neq \bar{y} \end{aligned} \tag{1.1}$$

where y and $\bar{y}$ are respectively the prediction results of feeding x and x^{adv} to f, and $\|\cdot\|_p$ denotes the L_p-norm.

The adversarial perturbations in an image AE are usually subtle in order to be human-imperceptible. To quantitatively describe such adversarial perturbations, L_p norms are usually used to measure the discrepancy between x and x^{adv}. According to the value of p in Equation 1.1, the mainstream AE generation algorithms can be categorized into three families: L_0, L_2 and L_∞ attacks. Informally, L_0 measures the number of modified pixels, L_2 the Euclidean distance between x and x^{adv}, and L_∞ the largest modification among all the modified pixels.

Depending on the manner of how $\bar{y}$ misleads a pre-trained classifier, adversarial attacks to neural networks can be categorized as either targeted or non-targeted. The aim of non-targeted attacks is to make the image be classified as any arbitrary class except the true one. By contrast, in targeted attacks the prediction result will be misguided to a specific class different from the correct one and desired by the attacker. Many image AE generation methods have been proposed and multiple off-the-shelf tools are available [6–9].

To defeat attacks based on AEs, both detection and defensive techniques attract the research community's attention. Given an input image, the ***detection*** system outputs whether it is an AE, so that the target neural network can reject those adversarial inputs. A ***defense*** technique, given an AE, helps the target neural network make correct prediction by either rectifying the AE or fortifying the classifier itself. In this work, we focus on AE detection and aim at accurate, resilient, and explainable countermeasures against AEs through exploiting their noticeable characteristics.

1.2 Adversarial Examples Generation

In this section, we will describe several popular AE generation methods briefly.

1.2.1 Fast Gradient Sign Method (FGSM)

As a classic non-targeted attack, FGSM is proposed to generate an adversarial example through adding a pixel-wise perturbation of magnitude [6]. In detail, the perturbation is computed as:

$$x^{adv} = x + \epsilon \cdot \text{sign}(\nabla_x J(x, y)) \tag{1.2}$$

where J is the classification loss function which has been used for training a target DNN model, and ϵ is a parameter that controls the the strength of perturbation. Since the perturbation for each pixel is performed only once along one gradient direction, FGSM is a very efficient way to generate AEs.

1.2.2 Iterative Gradient Sign Method (IGSM)

By implementing FGSM in an iterative way, Kurakin et al. [8] proposed IGSM which takes N small steps of magnitude $\alpha = \epsilon/N$ and adjusts the direction after each step. Furthermore, during computing perturbation, this algorithm clips the pixel values to ensure that they are in an reasonable range. The iterative computation of AEs using IGSM can be expressed as:

$$\begin{aligned} x_0^{adv} &= x \\ x_{i+1}^{adv} &= x_i^{adv} + \text{clip}(\alpha \cdot \text{sign}(\nabla_x J(x_i^{adv}, y))) \end{aligned} \tag{1.3}$$

where x_i^{adv} and clip respectively denote the perturbed sample at i iteration and a clipping of the adversarial sample's values. Since this iterative way makes the attack tend to overfit to a particular model, IGSM is more effective to white-box in which the target DNN is known than black-box i.e. attackers have no knowledge about that target model.

1.2.3 Jacobian Saliency Map Attack (JSMA)

The JSMA is a targeted attack based on a greedy iterative idea proposed by Papernot et al. [7]. It takes L_0 distance minimization as the optimization target, that is, the number of pixels that can be updated in the original image is bounded. To determine which pixels will be manipulated, the authors introduce the concept of *saliency map* which provides an adversarial saliency score for each pixel. One single pixel that possesses a higher adversarial saliency score usually has more impact on misleading the target model to predict a specific label desired by attackers. Thus, the attacker only manipulates those pixels that have high adversarial saliency scores in each iterative step based on a greedy strategy. The adversarial saliency score for each pixel is calculated as:

$$x_{i,t}^{adv} = x_{i,t} + \begin{cases} 0, & \text{if } \frac{\partial f_t(x)}{\partial x_i} < 0 \text{ or } \sum\limits_{j \neq t} \frac{\partial f_j(x)}{\partial x_i} > 0 \\ \frac{\partial f_t(x)}{\partial x_i} | \sum\limits_{j \neq t} \frac{\partial f_j(x)}{\partial x_i} |, & \text{otherwise} \end{cases} \tag{1.4}$$

where i denotes the ith pixel in the image, and f_j is the prediction value of the neuron j in the target model's output layer.

1.2.4 DeepFool

Moosavi et al. [10] developed the DeepFool attack that is used to create non-targeted AEs. The algorithm utilizes an iterative linearization of the classifier to generate L_2 minimization-based perturbations. To simplify the problem, the neural networks are imagined to be linear, so that the decision boundaries are a set of hyper-planes. Consequently, a polyhedron can be used to describe the output space. Assuming that f is a binary differentiable classifier, to mislead the decision of f near the current point x_i, the minimal perturbation is the orthogonal projection of x_i onto the separating hyper-plane. At each iteration the minimal perturbation of the linearized classifier is

computed as

$$\arg\min_{\delta_i} \|\delta_i\|_2 \quad \text{s.t. } f(x_i) + \nabla f(x_i)^T \delta_i = 0 \tag{1.5}$$

where δ_i is the perturbation imposed on x_i. Note that neural networks are not actually linear, so the search is repeated until a successful AE is found.

1.2.5 Carlini & Wagner Attacks

Carlini and Wagner [9] designed a group of targeted AE generation methods which are denoted as CW attacks. According to the distance metrics adopted in an optimization target, CW attacks can be divided into three types: L_0-, L_2- and L_∞-norm. The attack can be formulated as the following optimization problem:

$$\min \|\delta\|_p + c \cdot g(x+\delta) \quad s.t. \;\; x + \delta \in [0,1]^n \tag{1.6}$$

where $\|\delta\|_p$ is distance measurement. The function $g(\cdot)$ indicates whether the attack succeed or not, which is defined as:

$$g(x') = \max(Z(x')_{l_x} - \max\{Z(x')_i : i \neq l_x\}, -\kappa) \tag{1.7}$$

where $Z(\cdot)$ denotes the logits, that is the output of the last layer (before the softmax layer) of a DNN, and κ is a parameter that controls the *confidence-level* in creating an AE.

Due to a few creative designs, the CW attacks achieve performance superior to other attack methods. The first and foremost innovative design is using a logits-based objective function rather than softmax-cross-entropy loss, which plays a key role in the resilience improvement of the attack against defensive distillation [11]. Secondly, this algorithm maps the target variable to a space of the inverse trigonometric function, so that the problem is suitable to be solved by a modern optimizer, e.g. Adam [12]. Finally, a confidence-level parameter κ is introduced; as κ increases, the model classifies the resulting AE as the attacker-desired label more likely, giving

the attacker flexibility to make a trade-off between the degree of perturbations and misclassification probability.

1.3 Threat Model

The adversary has full knowledge of the target model (including both its architecture and parameters). He also knows the existence and internal details of the detector, and is allowed to *adapt attacks*. In adaptive attacks, the attacker tries to fool the image classifier and the detector at the same time. We consider adaptive attacks and evaluate the resilience of our detector to them in this work.

1.4 Structure of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 presents a novel and effective detector that tackles L_2 AEs through destroying the completeness of the influence by the perturbed pixels. In chapter 3, we consider the uncontrollable heavy perturbations as an inherent limitation of L_0 AEs. Based on this observation, we thwart such attacks by both detecting and rectifying them. Furthermore, a comprehensive hybrid detector which can detect all categories of AEs, i.e., L_0, L_2, and L_∞ attacks, is introduced in chapter 4. Finally, chapter 5 summarizes the proposed research, concludes this dissertation, and also discusses the future works.

Chapter 2

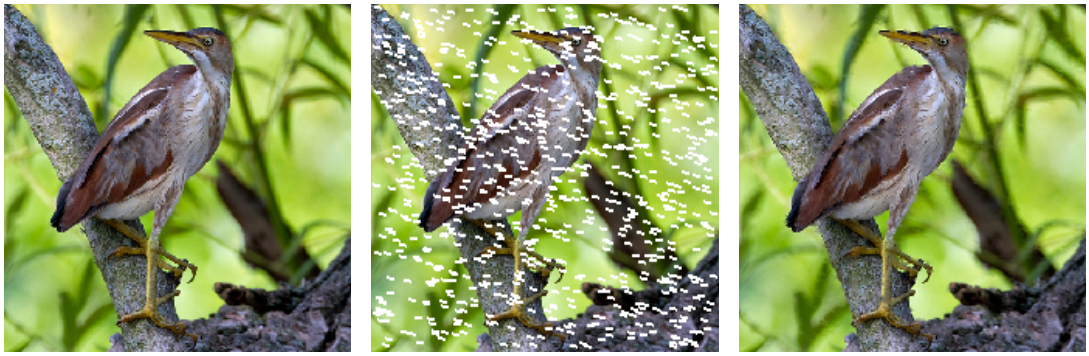
Exploiting the Sensitivity of L_2 Adversarial Examples to *Erase-and-Restore*

2.1 Introduction

L_2 adversarial perturbations by Carlini and Wagner (CW) are amongst the most effective but difficult-to-detect attacks. As suggested by Carlini and Wagner [9], defenders should consider evaluating "*a powerful attack*" and particularly emphasized L_2 attacks (Section 9 in [9]). Other researchers also agree that L_2 attacks by Carlini and Wagner (CW) [9] "*are among the most effective white-box attacks and should be used among the primary attacks to evaluate potential defences*" [13]. Although researchers have proposed many AE detection methods [14–17], recent studies [18–20] show that the detection usually goes ineffective when facing adaptive CW-L_2 AEs. Thus, how to accurately detect adaptive L_2 AEs is still an open question. We focus on tackling L_2 AEs in this chapter, and our goal is a technique that not only detects L_2 AEs accurately but is also resilient to adaptive attacks.

We have two key insights. First, we observe that those deliberately corrupted pixels exert a malicious influence *altogether* (e.g., through multiple rounds of optimizations during AE generation). It implies that a destruction of the completeness of the influence by the perturbed pixels can cause a failure of the attack. Second, while destruction may also harm the classification accuracy for benign samples, there exist very effective *inpainting* techniques [21–23] in the image processing area that can help restore a partially corrupted image. For example, Figure 2.1(a) shows an

original image, and Figure 2.1(b) a corresponding corrupted image where many regions are erased. After inpainting, as shown in Figure 2.1(c), the corrupted image is well restored.

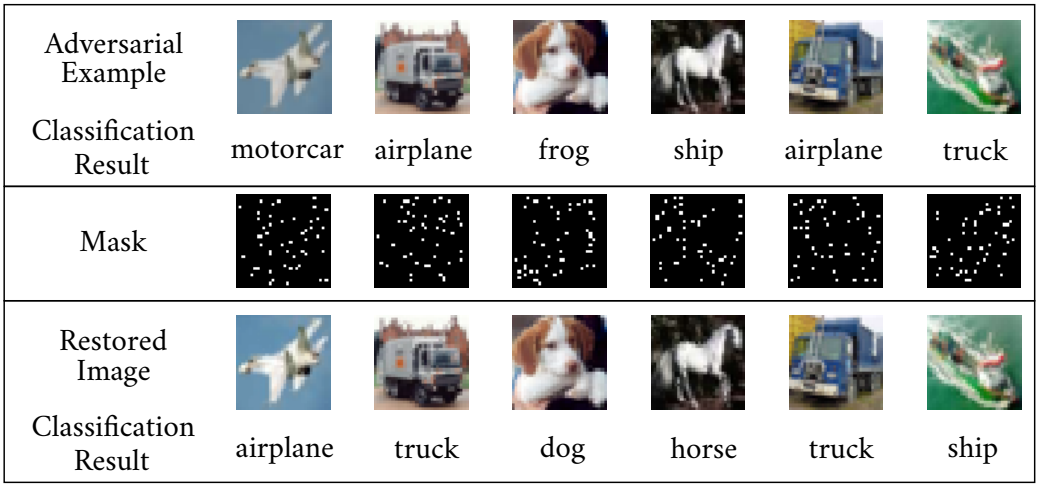


(a) Original image (b) Corrupted image (c) Restored image

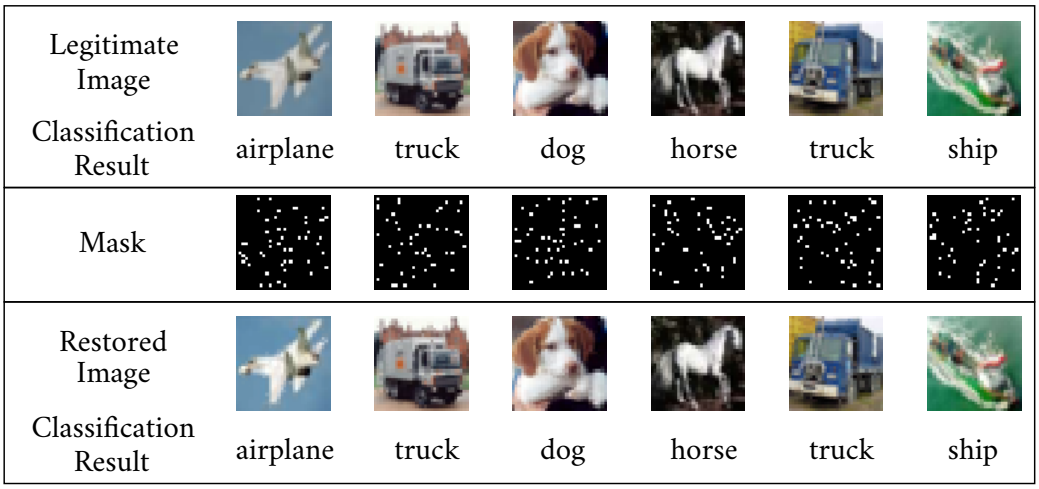
Figure 2.1: Restoring lost parts of an image with inpainting.

Thus, we hypothesize that if we ***randomly*** erase a portion of pixels from an AE and then apply inpainting to it, the attack will probably fail for two reasons. Discarding many small regions from an AE will ruin the holistic adversarial influence formed by the maliciously perturbed pixels. Second, the inpainting typically restores the image in a benign way that does *not* preserve the malicious influence. By contrast, if we apply the same "*Erase-and-Restore*" (E&R) operations to a benign sample, the classification results, before and after the steps, tend to be similar, as inpainting by design is to reverse deterioration of benign images.

Figure 2.2 illustrates our insights and observations using six color images from CIFAR-10. A *random mask* (mask, for short) in our work describes the locations of pixels that are randomly erased. We randomly erase 5% of the pixels of each image. The AEs are generated using the CW algorithm [9]. As shown in Figure 2.2(a), the classification results of each AE, before and after the E&R operations, are different. By contrast, as shown in Figure 2.2(b), the classification results of each benign sample, before and after the steps, are the same. Our large-scale experiments (Section 2.3) also show consistent results.



(a) Adversarial examples



(b) Benign samples

Figure 2.2: Different impacts of "Erase-and-Restore" on AEs and benign samples.

We consider the sensitivity to E&R operations as an exploitable characteristic of L_2 AEs, and propose a novel AE detection technique: given an image, if the classification results before and after E&R vary greatly, it is an AE; otherwise, a benign sample. We accordingly implement an L_2 AE detector, named THEMIS. To improve the detection accuracy, it is enhanced by applying E&R multiple times. Specifically, given an image I_0, we *randomly* erase some pixels of I_0 each time to create a sequence of images $\{I_1, I_2, \cdots, I_n\}$. Next, an inpainting technique is applied to them to obtain the restored images $\{I'_1, I'_2, \cdots, I'_n\}$. Finally, a classifier makes use

of the prediction results of I_0 and the restored images to determine whether I_0 is an AE.

We have evaluated our system using the popular image datasets CIFAR-10 and ImageNet. Two widely-discussed L_2 AE generation methods, CW [9] and DeepFool [10], are considered in the evaluation. We lay special emphasis on CW [9] because it can circumvent all existing detectors, especially when adaptive attacks are considered. Our experiments show that the proposed detection technique is very effective. Take the CW [9] attack as an example, on the CIFAR-10 dataset, THEMIS can detect 100% AEs with a false positive rate (FPR)=0, and on ImageNet, it can detect 99.3% AEs with FPR = 2.7%. In addition, the detection technique demonstrates three notable characteristics. ❶ It is **target-model agnostic**: a detector trained using AEs targeting one neural network model can be directly used to detect AEs targeting another. ❷ It has good **transferability**: a detector trained using AEs generated by one attack method can be directly used to detect AEs by another. ❸ More importantly, it shows **high resilience to adaptive attacks**. Finally, we interpret the effectiveness of the detection technique through both visualization and quantification.

To summarize, in this chapter, we find an interesting characteristic of L_2 AEs, whose classification results vary sharply when Erase-and-Restore operations are applied; meanwhile, benign samples are not so sensitive. Furthermore, we propose to exploit the characteristic for AE detection, and employ the idea of sampling to enhance the detection. By applying E&R for multiple times, richer features are generated to improve the detection accuracy. Besides, we implement the detection technique in THEMIS and evaluate it on two popular datasets, CIFAR-10 and ImageNet. The experiment results show that THEMIS outperforms prior techniques (such as NIC [24], LID [25], and Feature Squeezing [17]), achieving not only **the highest detection rate** but also the **lowest false positive rate**. Plus, due to its simplicity, it is extremely easy to apply and deploy. The detection technique is target-model

agnostic and shows high transferability across different L_2 attack methods. Not only that, it demonstrates strong resilience to adaptive CW-L_2 attacks, filling a critical gap in AE detection. Finally, we interpret the effectiveness of the detection technique in multiple ways.

2.2 Experimental Setup

Before presenting our defense scheme, we introduce the image datasets and the corresponding target neural networks on which we verify our key insights and evaluate the proposed approach.

Image datasets. We generate AEs using two popular datasets: CIFAR-10 and ImageNet, both of which are widely used in image classification tasks. In particular, for ImageNet, we adopt the *ILSVRC2012* samples to keep consistent with the prior state-of-the-art AE detector [24].

Target neural network models. (1) For CIFAR-10, we use two neural networks as the target models: a 32-layered ResNet model [26] (denoted as *ResNet32*), and a model structure described in [9] (denoted as *Carlini*). We train these two target neural network models from scratch (the accuracies of the two models are 91.96% and 78.86%, comparable with those published in prior works [17, 24]). (2) For ImageNet we re-use a 50-layered ResNet model [26] provided in Keras [27] (denoted as *ResNet50*).

AE generation and data preparation. Like existing AE detection works, only images that are correctly classified by the corresponding target model are used to generate AEs in our experiments. To generate *targeted* AEs, we designate the *next* class as the target class, similar to many other AE detection works [17, 24, 28]. Only AEs that can successfully fool the target models are used in the evaluation. For ImageNet, we collect 30,000 legitimate images and create 30,000 AEs: DeepFool and CW-L_2 generate 15,000 AEs each. The number of CW-L_2 AEs with each given

confidence level (i.e.,κ= 0.0, 0.4, and 1.0) is the same, that is 5,000 for each sub-group. In the dataset, 80% of instances are used for training and the remaining 20% for testing, denoted as $\mathcal{D}_I$-`Train` and $\mathcal{D}_I$-`Test`, respectively. Similarly, for CIFAR-10, based on the types of target model, we have four dis-joint datasets, $\mathcal{D}_C$-`Carlini-Train`, $\mathcal{D}_C$-`Carlini-Test`, $\mathcal{D}_C$-`ResNet-Train`, and $\mathcal{D}_C$-`ResNet-Test`. The former two and the latter two datasets have the same size and data composition as $\mathcal{D}_I$-`Train` and $\mathcal{D}_I$-`Test`, respectively. All AEs are generated using the opensource tool `Foolbox` [29].

Inpainting algorithm. The inpainting algorithm we choose in this work is designed by Telea [22]. This inpainting algorithm needs to solve an *Eikonal* equation, which is rarely differentiable everywhere. Considering the inpainting algorithm is *not* fully differentiable, it results in a non-negligible obstacle for adaptive attackers.

The experiments were performed on a computer running the Ubuntu 18.04 operating system with a 64-bit 3.6 GHz Intel® Core(TM) i7 CPU, 16 GB RAM and a GeForce® GTX 1070 GPU.

2.3 The Proposed Approach

2.3.1 Our Insights

Effects of erasing (or adding noises) alone. Due to the optimization nature of AE generation methods like CW and DeepFool, maliciously manipulated pixels in an AE are deliberately selected and perturbed. Thus, each of the perturbed pixels plays a certain role in the attack. By *randomly erasing* many pixels of an input image, it is likely to corrupt some of the perturbed pixels or their surrounding pixels in an AE, rendering the attack ineffective.

In the case of *benign* samples, however, the erasing operation, which is equivalent to introducing random noises to images, will significantly degrade the accuracy of the classifier. The close correlation between the image quality and the accuracy of image

classification has been widely studied in previous works [30–32]. They mention that neural networks are susceptible to random noise distortions. For example, Costa et al. [31] point out that *"noises can hinder classification performance considerably and make classes harder to separate*."

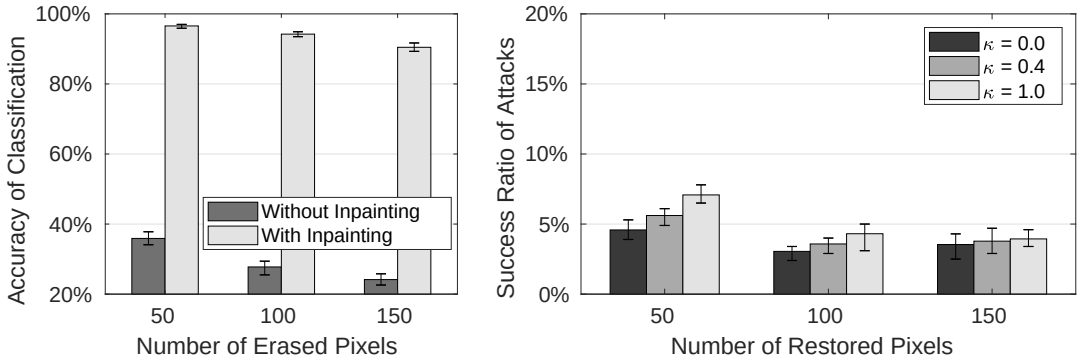
Combining erasing and inpainting. We thus propose to apply *inpainting* after the erasing operation. Inpainting is a category of techniques for restoring damaged regions of images. Given an erased region, an inpainting technique infers and recovers its original pixels. *Our insight* is that, while inpainting works very well for recovering benign samples, its recovering effect is usually *not* what the AE attacker desires, as the maliciously perturbed regions, once erased, can hardly be recovered to the attacker-intended values.

We further design experiments to verify the two insights in Section 2.3.2.

2.3.2 Insights Verification

From CIFAR-10, we randomly select 1,000 images that can be correctly classified by *ResNet32*. As shown in Figure 2.3(a), after randomly erasing 50∼150 (around 5%∼15%) of the pixels in each image, without inpainting, the classification accuracy significantly degrades from 100% to the range from 24.2% (when erasing 15%) to 35.9% (when erasing 5%), which verifies that erasing alone harms the classification accuracy for benign images significantly. By contrast, with inpainting applied, the classification accuracy recovers to 90.5%∼96.6%.

Besides, for each benign image we use the CW algorithm to generate three AEs with three different confidence levels (κ = 0.0, 0.4, and 1.0, respectively). All the AEs successfully fool the *ResNet32* model. As shown in Figure 2.3(b), after randomly erasing 50∼150 (around 5%∼15%) of the pixels in each AE and then restoring them using inpainting, the success rate of attacks dramatically decreases from the original 100% to the range 3.1%∼7.1%.



(a) Benign samples (b) Success rates of AEs with E&R

Figure 2.3: Impacts of E&R on benign samples and AEs.

Similar results can be observed on the ImageNet dataset as well. (1) Specifically, we randomly select 1,000 images from ImageNet that can be correctly classified by the *ResNet50* model. For example, after erasing and restoring 5% of the pixels in each image, the classification accuracy stays at 96.3%. (2) On the other hand, when we apply the same erasing and restoring operations to the 1,000 AEs generated from these benign images, the success rate of attacks decreases from 100% to around 4.1%.

Therefore, it can be concluded that E&R has very small impacts on benign samples, but large impacts on AEs, demonstrating a noticeable contrast.

2.3.3 Approach Details

Based on our insights, we propose a novel AE detection technique, named E&R, that exploits the sensitivity of AEs to E&R operations, and implement it in a system, called Themis, as shown in Figure 2.4. (1) Given an input image I_0, we ***randomly erase*** λ ***pixels*** of it to create a deteriorated image I. Employing the idea of sampling, this step is repeated for n times to obtain a sequence of deteriorated images $\{I_1, I_2, \cdots, I_n\}$. The *intuition* behind it is that even if an AE "luckily" evades the detection once, it is very unlikely for it to hide itself throughout the multiple samples. (2) Next, an inpainting technique is leveraged to produce a corresponding sequence of ***restored*** images $\{I'_1, I'_2, \cdots, I'_n\}$. (3) Finally, we feed both the input image I_0

and $\{I'_1, I'_2, \cdots, I'_n\}$ into a neural-network classifier, and collect all the classification results.

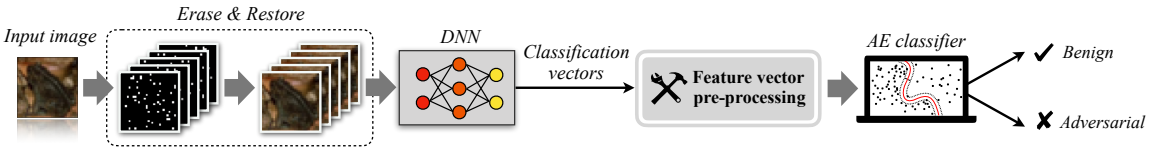


Figure 2.4: Architecture of THEMIS.

Given an image in CIFAR-10, its classification result is a vector $\in \mathbb{R}^{10}$ (since there are 10 classes in the dataset). We simply concatenate all the classification-result vectors for both I_0 and $\{I'_1, I'_2, \cdots, I'_n\}$ to obtain a feature vector $\in \mathbb{R}^{10\times(n+1)}$ for training the AE classifier.

Given an image from the ImageNet, its classification result is a vector $\in \mathbb{R}^{1000}$ (since there are 1,000 classes in the dataset). Thus, the number of features to be fed to our classifier is $1000 \times (n+1)$, which is too large. To make the training of our classifier more feasible, Principal Component Analysis (PCA) is performed on the classification results of I_0 and $\{I'_1, I'_2, \cdots, I'_n\}$, to reduce the dimensionality to a lower value d. Unless otherwise specified, we set d to 10 (1% of the original dimensionality) to keep consistent with CIFAR-10. Note that the number of principal components should be less than both the number of features and the number of samples, when solving PCA based on the truncated SVD (singular value decomposition). In our case, the number of samples is $n+1$; we thus let $n = 11$ (we discuss the impact of n's values with detailed experimental results in Section 2.4.3). We concatenate the vectors of principal components for both I_0 and $\{I'_1, I'_2, \cdots, I'_n\}$ to obtain a feature vector for training our classifier.

The value of the parameter λ (number of pixels to be erased) is set to 10% of the pixels in an input image. We adopt this value for two reasons. (1) As shown in Figure 2.3, when 10% of the pixels are erased and restored, it harms the success rate of AEs most heavily, without degrading the classification accuracy for benign samples

significantly. (2) The inpainting algorithm we adopt performs very well when the portion of corrupted pixels in an image is less than 15% [22].

It is worth mentioning that $\lambda = 10\%$ leads to an enormous randomness pool. Take an image in CIFAR-10 as an example, the size of which is 32×32: with λ=100 ($\approx$ 10% of the pixels), the number of unique masks is around 7.7×10^{140}. It is thus very unlikely for an adaptive attacker to correctly predict which masks will be used by our detector.

We train our AE classifier using two supervised learning techniques: *AdaBoost* [33] and *SVM* [34].

2.4 Evaluation

We evaluate the detection performance of the proposed scheme against L_2 attacks in terms of *detection rate* and *false positive rate* (FPR). The detection rate is defined as the ratio of the number of successfully detected AEs to the total number of AEs. FPR refers to the fraction of benign samples that are misclassified as AEs.

2.4.1 Detection Performance

We use $\mathcal{D}_I$-`Train`, $\mathcal{D}_C$-`Carlini-Train`, and $\mathcal{D}_C$-`ResNet-Train` (see Section 2.2) to train our detectors and evaluate them based on the corresponding testing sets.

CW-L_2 attacks. As shown in Table 2.1, the proposed technique achieves very high detection rates (up to 100% on CIFAR-10, and 99.3% on ImageNet) with low FPR values. The results are stable across different target models, confidence levels, and classification methods.

In addition to SVM and Adaboost, we also train a fully connected neural network as the AE classifier, and obtain very similar results. It shows that it does not affect the performance by using a more sophisticated classifier. It also indicates that the effect of E&R does not depend on a specific classifier type.

Table 2.1: Performance of THEMIS.[1]

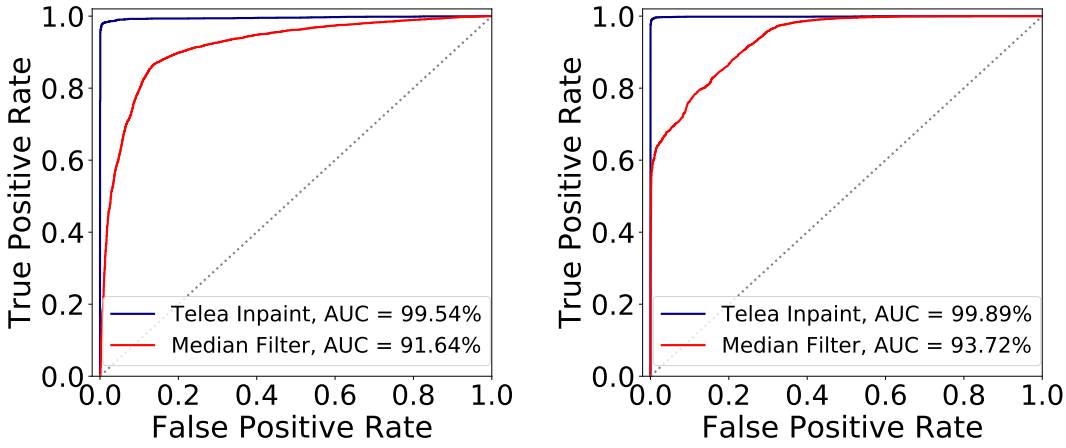
Dataset	Target Model	Classifier	FPR	DR: CW-L_2			DR: DeepFool
				κ=0.0	κ=0.4	κ=1.0	
CIFAR-10	Carlini	SVM	0.6%	100%	100%	100%	99.4%
		AdaBoost	0.0%	100%	100%	100%	98.3%
	ResNet32	SVM	2.8%	99.4%	99.6%	99.6%	99.8%
		AdaBoost	0.9%	99.4%	99.2%	99.4%	99.8%
ImageNet	ResNet50	SVM	3.5%	97.9%	98.4%	98.7%	93.7%
		AdaBoost	2.7%	98.9%	99.2%	99.3%	95.0%

[1]After THEMIS is trained using training datasets that contain benign samples, CW and DeepFool AEs, the DR (Detection Rate) and FPR (the rate of benign samples misclassified as AEs) are measured using testing sets.

DeepFool attacks. For another leading L_2 AE generation algorithm—DeepFool (see Section 1.2.4), we observe very similar results as CW-L_2. Table 2.1 shows that our detector achieves very high detection rates (up to 99.8% on CIFAR-10, and 95.0% on ImageNet) with low FPR values.

Comparison with baseline. To illustrate the the benefits of the Telea inpainting algorithm used in our detector, we compare it with a baseline method, which uses a median filter to recover the damaged pixels. In particular, the window size of our median filter is 3×3, which is also adopted by Feature Squeezing [17]. Without loss of of generality, the datasets we use are $\mathcal{D}_C$-`ResNet-Train` and $\mathcal{D}_C$-`ResNet-Test`. We replace the Telea inpainting with the median filter in our implementation to build a baseline detector. Figure 2.5 shows the comparison result using ROC (receiver operating characteristic) curves of the different detectors. As shown in Figure 2.5(a), when SVM is used as the classifier, the AUC value declines from 99.54% to 91.64%. Similarly, as shown in Figure 2.5(b), when AdaBoost is used, the AUC value correspondingly declines from 99.89% to 93.72%. Thus, a high-quality inpainting method is closely related to the final performance of our AE detector.

Comparison with prior work. As summarized in Table 2.2, we compare THEMIS with some state-of-the-art AE detectors—NIC [24], LID [25], and Feature Squeez-



(a) SVM (b) AdaBoost

Figure 2.5: ROC curves.

ing [17]. For CW-L_2 attack, their experiments only examine $\kappa = 0.0$, which is the default setting, so we also list the results under $\kappa = 0.0$ in Table 2.2 (see Table 2.1 for the results of our detector under other κ values). We take NIC as an example here. With respect to CIFAR-10, NIC obtains the detection rate 96% (see Table I in [24]), while our system achieves the detection rate **100%**. With respect to ImageNet, the detection rate of NIC is 96% (see Table I in [24]), while our detection rate is **98.9%**. In terms of DeepFool, THEMIS also outperforms other AE detectors. When considering CIFAR-10, our system obtains the detection rate **99.4%**, while NIC [24] obtains the detection rate 91.0% (see Table I in [24]). Similarly, when considering ImageNet, THEMIS can achieve the detection rate **95.0%**, that is superior to NIC, the detection rate of which is 92%.

Table 2.2: Comparison with other AE detectors (DR: Detection Rate).[1]

Dataset	CIFAR-10				ImageNet			
Detector	THEMIS	NIC	FS	LID	THEMIS	NIC	FS	LID
FPR	0.6%	4.2%	5.6%	4.9%	2.7%	14.6%	8.3%	14.5%
DR: CW-L_2	100%	96%	100%	86%	98.9%	96%	92%	78%
DR: DFool	99.4%	91%	77%	84%	95.0%	92%	79%	83%

[1]We use the same attack settings as used in prior work [17, 24].

More importantly, from the angle of FPR, the performance of THEMIS is significantly better than other detectors. For example, when considering CIFAR-10, the FPR of NIC is 4.2%, while ours is **0.6%**. Moreover, when considering ImageNet, the FPR of NIC is 14.6%, while ours is only **2.7%**. It is worth noting that the distribution of adversarial and benign images is not balanced in practice—most inputs should be benign. Thus, FPR is a very important metric to evaluate the model performance: a lower FPR indicates that the system makes fewer mistakes for benign images. THEMIS is able to keep both a high detection rate and a *very low FPR*.

2.4.2 NOTABLE CHARACTERISTICS

Target-model agnostic. We are interested in finding out whether a detector trained using AEs targeting one model can be directly used to detect AEs targeting another—that is, whether it is *target-model agnostic*. We thus train our system using CW-L_2 AEs in $\mathcal{D}_C$-`Carlini-Train`, and test it using CW-L_2 AEs in $\mathcal{D}_C$-`ResNet-Test`.

Table 2.3: Target-model agnostic property of THEMIS.

Target Model (Train → Test)	**Classifier**	**Detection Rate**		
		κ=0.0	κ=0.4	κ=1.0
Carlini→ResNet32	SVM	100%	100%	100%
	AdaBoost	97.9%	97.9%	98.2%
ResNet32→Carlini	SVM	99.9%	99.9%	99.8%
	AdaBoost	99.7%	99.8%	99.6%

As Table 2.3 shows, the detection rate is as high as 100%. We then train the system using CW-L_2 AEs in $\mathcal{D}_C$-`ResNet-Train`, and test it using CW-L_2 AEs in $\mathcal{D}_C$-`Carlini-Test`; the detection rate is as high as 99.9%.

Therefore, this experiment not only confirms that THEMIS is *target-model agnostic*, but also demonstrates that THEMIS has low risk of overfitting.

Transferability. We are also interested in the transferability of our detector—whether THEMIS trained on one type of AEs can be directly applied to detect another

type of AEs that are *unseen* during training. To verify it, we *train* THEMIS using CW-L_2 AEs in $\mathcal{D}_C$-`Carlini-Train`, without loss of generality. Then, we test the trained system using DeepFool AEs in $\mathcal{D}_C$-`ResNet-Test` and $\mathcal{D}_C$-`Carlini-Test` , and our system can achieve detection rates 97.1% and 96.2%, respectively. Thus, we can conclude the proposed technique has very good transferability, that is, it keeps effective in handling unseen AE generation methods.

Explanation. The two notable properties of THEMIS—target-model agnostic and good transferability—can be attributed to the unique advantage of the proposed approach: benign samples and AEs show distinct sensitivities to the E&R operations, which do not depend on the target model and the attack method.

2.4.3 VALUE SELECTION FOR THE PARAMETER n.

We use $n = 11$ in the previous experiments. Here, we investigate the impacts of different values of n on the detector's performance. The CW-L_2 AEs in $\mathcal{D}_C$-`Carlini-Train`, $\mathcal{D}_C$-`ResNet-Train`, and $\mathcal{D}_I$-`Train` are used in this experiment. For CIFAR-10, which has only 10 classes (thus no PCA is needed), varying the value of n has little impacts. However, for ImageNet, the value of n has noticeable impacts: when n increases, the AE detection rate increases and FPR decreases (see Table 2.4 and Table 2.5 for more details). The reason is that by increasing n, more principal components can be extracted (see Section 4). However, when $n > 11$, the performance improvement is negligible, probably because the extra principal components do not provide useful features for AE detection. Therefore, we adopt $n = 11$.

2.4.4 EFFICIENCY OF THEMIS

We investigate the efficiency of the proposed technique on ImageNet because large-sized images consume more processing time. For a single image, ResNet50 needs approximately 1.076 seconds for classification. Since parallel computing is supported

Table 2.4: Impacts of different values of n (CIFAR-10).

Target Model	**Classifier**	**FPR**	**Detection Rate**			
			κ=0.0	κ=0.4	κ=1.0	
Carlini	SVM	0.4%	100%	100%	100%	n=3
	Adaboost	0.0%	100%	100%	99.9%	
ResNet32	SVM	3.6%	99.6%	99.6%	99.6%	
	Adaboost	0.9%	99.2%	99.1%	98.5%	
Carlini	SVM	0.4%	100%	100%	100%	n=5
	Adaboost	0.0%	100%	99.9%	99.9%	
ResNet32	SVM	3.3%	99.6%	99.6%	99.6%	
	Adaboost	0.7%	99.2%	99.1%	99.0%	
Carlini	SVM	0.4%	100%	100%	100%	n=7
	Adaboost	0.0%	100%	100%	99.8%	
ResNet32	SVM	2.9%	99.6%	99.7%	99.7%	
	Adaboost	0.9%	99.3%	99.1%	98.9%	
Carlini	SVM	0.4%	100%	100%	100%	n=9
	Adaboost	0.0%	100%	100%	99.8%	
ResNet32	SVM	3.0%	99.7%	99.7%	99.7%	
	Adaboost	0.7%	99.3%	99.3%	99.1%	
Carlini	SVM	0.4%	100%	100%	100%	n=11
	Adaboost	0.0%	99.8%	99.9%	99.7%	
ResNet32	SVM	2.8%	99.6%	99.7%	99.7%	
	Adaboost	0.8%	99.0%	99.2%	98.9%	

Table 2.5: Impacts of different values of n (ImageNet).

Target Model	**Classifier**	**FPR**	**Detection Rate**			
			κ=0.0	κ=0.4	κ=1.0	
ResNet50	SVM	9.8%	95.4%	95.1%	95.5%	n=3
	Adaboost	6.6%	93.1%	91.4%	93.8%	
	SVM	4.7%	95.5%	95.8%	97.3%	n=5
	Adaboost	2.8%	96.5%	97.6%	97.2%	
	SVM	3.6%	97.6%	98.1%	98.2%	n=7
	Adaboost	2.1%	97.9%	98.6%	98.6%	
	SVM	3.5%	97.6%	98.0%	98.3%	n=9
	Adaboost	2.0%	98.0%	98.4%	98.8%	
	SVM	3.2%	97.6%	98.1%	98.5%	n=11
	Adaboost	1.4%	98.4%	98.5%	98.9%	

by GPU, given a relatively small number of images as inputs (e.g., $n = 11$), it takes similar time to generate the classification vectors for them. Apart from this, to detect AE, our method brings additional 1.01 seconds by average. In detail, it consumes 0.264 seconds for the inpainting, 0.744 seconds for the PCA-based dimension reduc-

tion, and 0.002 seconds for the final prediction (taking SVM as an example). In short, our detector causes a small delay.

2.5 Resilience to Adaptive Attacks

In an adaptive attack threat model, an adversary knows the existence and internal details of our detector and *adapts* the attacks to bypass the detection. We thus seek to study the resilience of Themis to adaptive attacks.

An AE detector can be categorized as either differentiable or non-differentiable. Several previous works propose defense mechanisms that apply differentiable transformations to an image before detection or classification [15, 35–37]. But attackers can circumvent these differentiable defenses by "*differentiating through them*"—*i.e.,* by taking the gradient of a class probability regarding input pixels through both the CNN and the transformation [18, 20, 38]. This strategy, however, is *inapplicable* to bypassing Themis. Due to the random-erasing and inpainting-based restoring, our approach is not only non-differentiable but involves tremendous randomness.

To bypass non-differentiable defences, Backward Pass Differentiable Approximation (BPDA) is proposed [39]. To handle defenses that employ randomized transformation to the input (like ours), it applies Expectation over Transformation [40] to compute the gradient over the expected transformation to the input. However, in our approach the erased pixels are randomly selected among all the image pixels, and there are around 7.7×10^{140} unique masks (even for a small image; see Section 2.3.3); thus, it is infeasible to calculate the expected transformation. Moreover, Themis is not only randomized but also non-differentiable; in this case, it is unknown how to apply BPDA to bypassing Themis.

Adaptive AE generation. He et al. [18] describe a representative adaptive attack method against non-differentiable defences, where an attacker tries to circumvent the

defensive approach by (a) considering intermediate distorted images during optimization and (b) exploring multiple diverse optimization paths.

Inspired by [18], we design similar adaptive attacks to examine the resilience of our approach. To that end, we modify the code of the CW algorithm [9], in order to adaptively generate AEs that can bypass our detector. Specifically, after each iteration in an optimization procedure, an intermediate distorted image is obtained. We then check whether it can bypass our detector. For each image, we repeat the optimization procedure for up to T times to explore different optimization paths (for this purpose, we set a randomly initialized state at the beginning of each optimization procedure). As shown in Figure 2.6, we set $T = 150$, corresponding to around 450 seconds on average on our machine. In comparison, the two works [37] and [18] use around 75 and 180 seconds to generate adaptive AEs for each image, respectively.

Given that adaptive CW AE generation is quite time-consuming, without loss of generality, this experiment is conducted on 500 images randomly selected from CIFAR-10. During the AE generation, we let $\kappa = 0.0$, which means that the resulting AE is classified as the target class. As κ increases, the model classifies the resulting AE as the attacker-desired label more likely. As a larger value of κ imposes an extra constraint to attackers and lowers the chance of successful adaptive attacks, we only consider $\kappa = 0.0$.

Resilience results. We adopt the SVM-based detector that achieves a detection rate of 100% (Table 2.1): no AEs can fool it *without adaptive attacks*. Figure 2.6 shows that only 4.2% (that is, 21 AEs) of adaptive AEs can bypass our detector. By contrast, similar adaptive attacks [18] can bypass *feature squeezing* based AE detection [17] at a success rate of 100%; as another example, [37] can merely achieve a detection rate of 70% under adaptive CW attacks. More importantly, the first 50 times of the optimization path exploration attain the success rate of 3.4%, while the following 100 times only increase the success rate by 0.8%. It shows that the effect

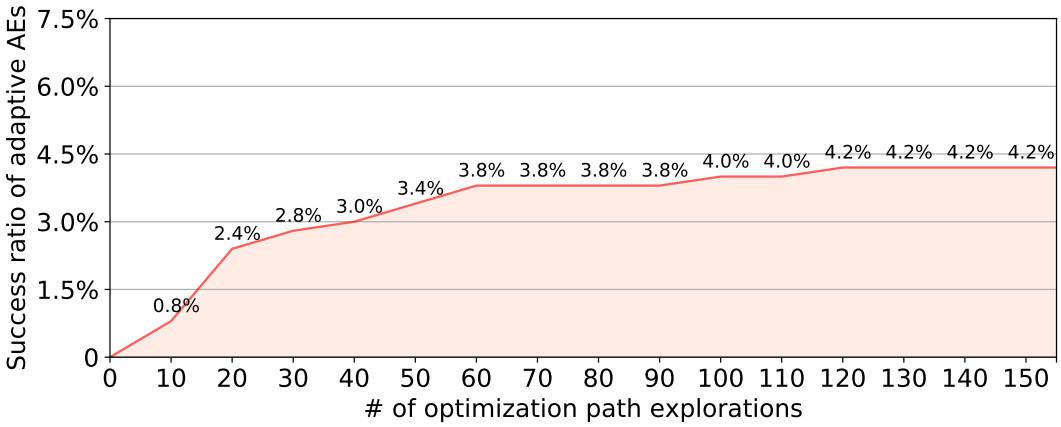


Figure 2.6: Success ratio of adaptive AEs.

of adaptive attacks grows very slowly as the attacker doubles his time. We thus can conclude that our detection technique is not only resilient to adaptive attacks based on differentiation, but also to adaptive attacks through exploration of many optimization paths. Thus, THEMIS, highly resilient to adaptive CW-L_2 attacks, fills a critical gap in AE detection.

2.6 INTERPRETABILITY

Background. To make the final prediction, most neural-network-based image classifiers implement a *softmax* function at the last layer

$$softmax(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \quad \text{for } i = 1, \cdots, K \text{ and } \mathbf{z} = (z_1, \cdots, z_K) \in \mathbb{R}^K \tag{2.1}$$

which maps an input vector $\mathbf{z}$ consisting K real numbers to a probability mass function over predicted output classes. The input vector of a *softmax* function is also called *logit*. Given a benign image whose logit is $\mathbf{z}$, the goal of an attacker is to perturb the image to get a new logit $\mathbf{z}'$ such that $\text{argmax}_i(\mathbf{z}') \neq \text{argmax}_i(\mathbf{z})$.

Interpretation Using Classification Results. Let $f(x)$ be the output of the *softmax* layer of a neural network f when feeding the input x. Let $T(x)$ be the

output of processing x with E&R operations. If x is benign, since it is not sensitive to E&R operations, the probability mass functions $f(x)$ and $f(T(x))$ are similar. By contrast, if x is an AE, $f(x)$ is significantly different from $f(T(x))$, since AEs are very sensitive to E&R operations. In short, if the sensitivity distinction between AEs and benign samples is true, the divergence (or distance) between $f(x)$ and $f(T(x))$ should reflect whether x is malicious or benign. We then adopt two widely used metrics, Wasserstein distance (WD for short) [41] and Kullback-Leibler divergence (KL for short) [42].

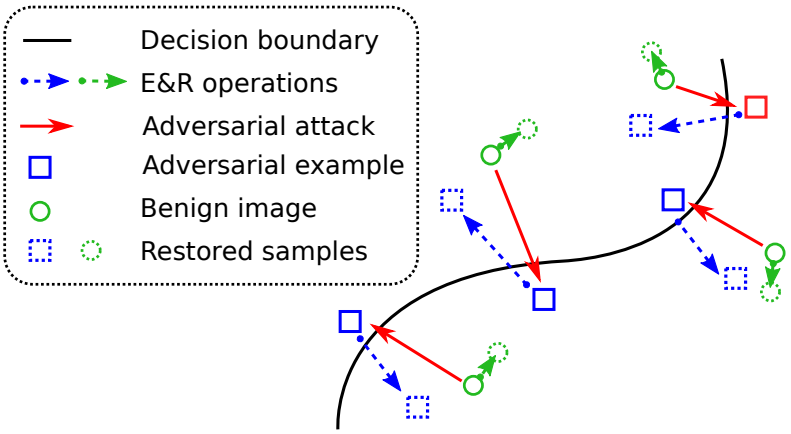


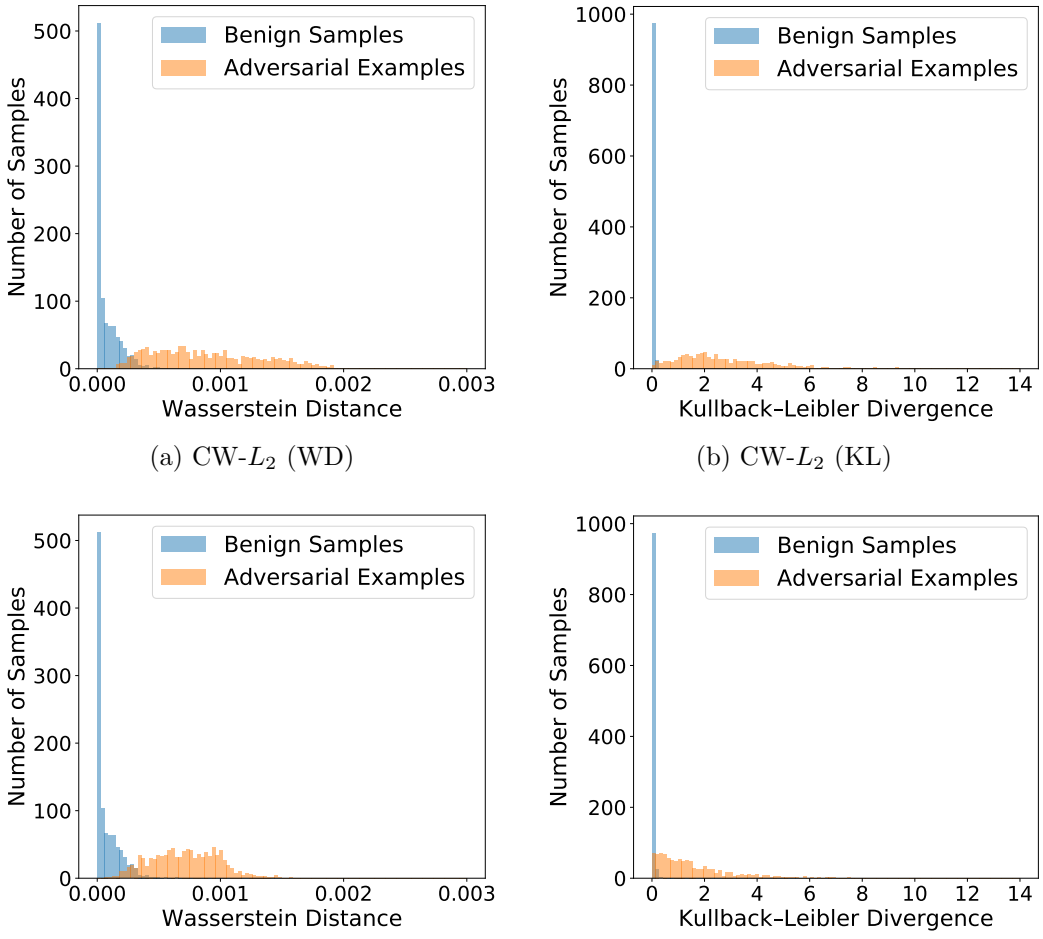
Figure 2.7: Illustration of how E&R works.

As shown in Figure 2.7, we depict benign and adversarial examples by green circles and blue squares, respectively. The arrows with dotted line represent E&R operations. We consider the changes caused by E&R operations on benign images and AEs (depicted by green and blue arrows with dotted line, respectively) should fall into different probability distributions. To visualize this, we randomly select 1,000 image pairs consisting of AEs and benign instances from $\mathcal{D}_I$-`Test`. After feeding them (with and without applying E&R operations) into the image classification model, we collect the output of the *softmax* layer. Then, we measure the difference between $f(x)$ and $f(T(x))$. To be consistent with the design of THEMIS, we apply E&R operations 10 times for each image and calculate an arithmetic mean of the 10 measurements.

The visualization of samples is shown in Figure 2.8, which confirms our proposition; that is, the changes caused by E&R operations on benign images and AEs fall into different clusters.

Table 2.6: Clusters splitting result.

Attacks	**Metrics**	**FPR**	**TPR**
CW-L_2	WD	0.5%	78.4%
	KL	0.0%	96.1%
DeepFool	WD	1.1%	85.7%
	KL	0.5%	89.3%



(a) CW-L_2 (WD)

(b) CW-L_2 (KL)

(c) DeepFool (WD)

(d) DeepFool (KL)

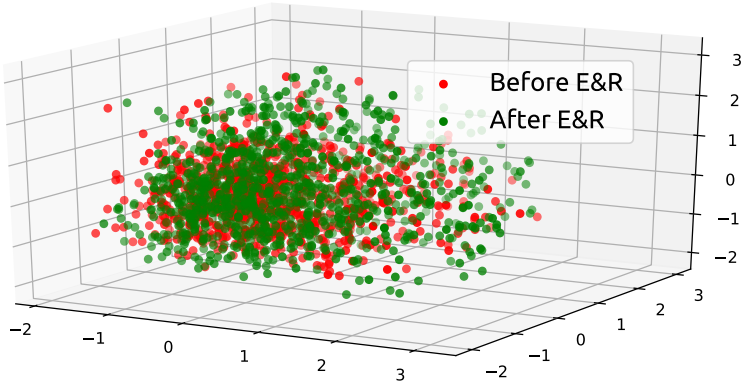
Figure 2.8: Visualization of the changes caused by E&R on benign samples and AEs.

Next, we quantitatively analyse to what extent the distance/divergence measurement can help discriminate an AE that is across the decision boundary. In detail, we use an optimal threshold based on the ROC (receiver operating characteristic) curve, to split AEs and benign images distributions. Table 2.6 presents the FPR and TPR (i.e., Detection Rate defined in Section 2.4). Note that the results are only for illustrating that E&R imposes different impacts on AEs and benign samples in terms of probability mass function changes, and do not represent the detection performance of Themis (see Section 2.4 for its detection performance). Here, we only use one dimensional feature (i.e., the Wasserstein distance or KL divergence) to split two clusters, information loss inevitably degrades the splitting performance, which is mitigated by the design of Themis.

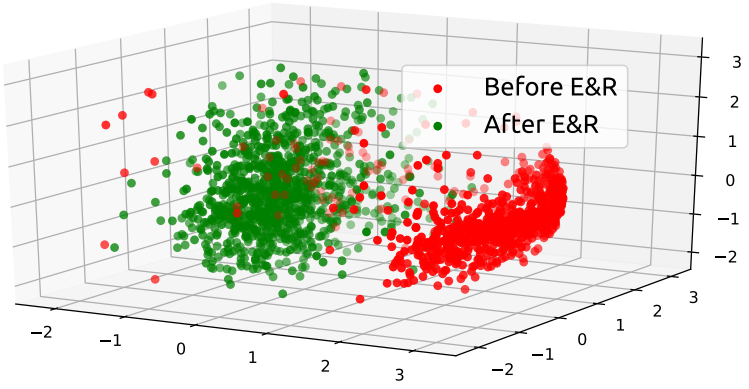
Interpretation through Visualization of Feature Vectors. The feature vectors due to 1,000 randomly selected benign samples from the ImageNet dataset and the corresponding 1,000 AEs are visualized in Figure 2.9. For the visualization purpose, it shows only three principal components of the pre-processed feature vectors (see Figure 2.4). We have two observations. (1) While the feature vectors of benign samples, before and after the E&R operations, are close (Figure 2.9(a)), those of AEs form two clusters far apart (Figure 2.9(b)). (2) PCA is effective in preserving features that help distinguish benign samples from AEs.

2.7 Related Work

Countermeasures against AE attacks can be roughly divided into two categories. The first category aims to eliminate the influences of AEs by either rectifying them or fortifying the target neural network itself. The second category is AE detectors (including our work), the goal of which is to predict whether an input is adversarial,



(a) Benign samples



(b) AEs

Figure 2.9: Visualization of feature vectors.[1]

so that the target neural network can reject those inputs. Given the large body of research on AEs, this is not intended to be exhaustive.

2.7.1 Adversarial Influences Elimination

To improve the robustness of neural networks, *adversarial training* augments the training set with the label-corrected AEs [43, 44]. Buckman et al. [45] propose using thermometer-encoded inputs to assist adversarial training. Alternatively, *Shield* [46] enhances a model by re-training it with multiple levels of compressed images using JPEG, a commonly used image compression technique.

[1]The coordinate axes respectively represent three largest principal components.

Another strategy is to pre-process the inputs before feeding them to neural networks. For instance, the pixel deflection and a wavelet-based denoiser are combined to rectify AEs [38]. Liao et al. [47] propose higher-level guided denoisers aiming to remove the adversarial noise from inputs. Some other methods adopt JPEG compression techniques [48, 49] to filter out the information redundancy, which otherwise provides living space for adversarial perturbations. However, their accuracies under adaptive attacks are lack of adequate evaluations. CIIDefence [50] proposes to use image inpainting with wavelet based denoising to rectify the classification result. However, its inpainting mask is guided by class activation maps, which can be predicted and exploited by an adaptive attacker. Both MagNet [16] and [51] essentially take the path of removing noises/enhancing images, rather than the Erase-and-Restore path proposed in this work. REMIX [51] applies inpainting to rectifying classification results, with an rectifying accuracy 86% on CIFAR-10. It uses autoencoder as the inpainter. Autoencoders are typically data-specific, which means that it is only effective on images similar to what they have been trained on. It did not study the resilient to adaptive attacks and did not provide interpretation either.

Unlike all these works, the purpose of our work is for highly accurate attack detection, e.g., an accuracy of over 98% on CIFAR-10 and ImageNet. It does not have dependency on high similarity between training data and testing data. It is target-model agnostic: a detector trained using AEs targeting one model can be directly used to detect AEs targeting another. Moreover, our work provides interpretation why the detection method works, and carefully examines its resilience to adaptive attacks.

2.7.2 Adversarial Examples Detection

Li et al. [14] extract PCA features after inner convolutional layers of the DNN, and then use a cascade classifier to detect AEs. Metzen et al. [15] train a CNN-based

auxiliary network. This light-weight sub-network works with the target model to detect AEs. Some techniques apply pre-processors on input images and use prediction mismatch strategy to detect AEs. For example, Meng et al. [16] train an auto-encoder as the image filter. If the predictions of an original image and the corresponding processed one fail to match, the input is adversarial. Similarly, Xu et al. [17] propose *feature squeezing* to detect AEs by comparing the prediction for the original input with that for the squeezed one. However, adaptive attacks have successfully circumvented all of the aforementioned detection methods [18–20]. Finally, Tian et al. [37] leverage image rotation and shifting as pre-processors to construct a detector. Although these operations can produce certain randomness to counter some adaptive attacks, their randomness pool is very limited. It only has 45 possible transformations. As a result, their method can merely achieve a detection rate of 70% under adaptive attacks [37].

Zeng et al. [52] proposes a novel AE detection method inspired by multiversion programming, which first uses multiple off-the-shelf audio recognition systems to classify the same audio input and then compares the classification results to detect AEs. Their insight is the extraordinary difficulty of generating highly transferable audio AEs, which is not the case for image AEs. We also make use of multiple classification results, which, however, is based on the idea of sampling (i.e., applying E&R multiple times) to enhance the detection accuracy.

Inpainting has been used in our prior AE detection work [28], but it was applied in a different way. Specifically, [28] focuses on detecting L_0 attacks by inpainting salient noises, as L_0 attacks usually cause large-amplitude perturbations due to minimizing the number of modified pixels.

The AE detection idea that *intentionally* and *randomly* "damages" (i.e., erases) some pixels of an image and then uses an inpainting algorithm is not only novel and effective, but can also be interpreted and keep resilient to adaptive attacks. Unlike other very complex methods, our method is extremely simple and easy to apply.

Although it only handles L_2 attacks, it can easily work as a plugin or complement to enhance an existing attack detection system.

2.8 Summary

Our finding has revealed that L_2 AEs are sensitive to the Erase-and-Restore operations, while benign samples are not. Exploiting the sensitivity distinction, we have proposed a novel and effective AE detection approach E&R. It outperforms other state-of-the-art approaches in terms of both high detection rates and low false positive rates. In addition, our detector is target-model agnostic, keeps effective across different L_2 attack methods (i.e., good transferability across attack methods), and is resilient to adaptive attacks. Furthermore, we have interpreted the detection technique from both qualitative and quantitative angles to provide deeper understanding of the technique. Unlike many other detection methods that are complex and thus difficult to construct and train, this method is very simple to build and easy to apply in practice.

Chapter 3

Exploiting the Inherent Limitation of L_0 Adversarial Examples

3.1 Introduction

In the last chapter, the proposed erasing and restoring approach works by destruction of the carefully perturbed pixels. Attackers thus may consider minimizing the number of perturbed pixels, like in L_0 AEs, to evade our detection. This, however, essentially becomes L_0-norm attacks, which are a category of widely discussed threats where adversaries are restricted in the number of pixels that they can corrupt. Although many AE detection methods [53–55] and defense techniques [46, 47, 56] have been proposed, prior methods either are not very effective in handling L_0 AEs or omit discussing them. For example, feature squeezing [17] is capable of detecting L_0 AEs. However, He et al. [18] have shown that feature squeezers, either single or joint, are not resilient to adaptive attacks. Previous work even argues that it is challenging to recover the correct classification of L_0 AEs by input transformation, as "*it is very difficult to properly reduce the effect of the heavy perturbation*" [54].

We identify two characteristics of L_0 AEs. By exploiting the two characteristics, we build a detector based on a very simple architecture that achieves a high detection accuracy. Moreover, a pre-processor based on these observations can effectively rectify L_0 AEs to recover the correct classifications.

The first characteristic is that it limits the number of modified pixels, but not the amplitude of pixels. Thus, L_0 attacks tend to introduce large-amplitude perturba-

tions, especially for targeted attacks that aim to achieve an attacker-desired output from a neural network. Second, as L_0 attacks try to modify as few pixels as possible, the optimization-based AE generation process tends to result in altered pixels that scatter in the image. In other words, those corrupted parts are mostly small and isolated regions. Both characteristics are verified by our experiments.

We accordingly propose a novel AE detection method. The main novelty is that we convert the *AE detection problem* into a *comparison problem*. Specifically, the architecture of the detector uses a Siamese network [57], which is known to be powerful in comparison. Given an image I, it is processed by a pre-processor to obtain another image I'. The Siamese network takes I and I' as the inputs and outputs whether I is an AE. The advantage of the design is that the Siamese network is able to automatically and precisely capture the discrepancies between the two inputs for AE detection.

Another advantage is that the pre-processor used for AE detection can also work as an effective defense by removing the influence of the adversarial perturbations. Specifically, we propose an *inpainting*-based algorithm to process images, where *inpainting* refers to the process of *reconstructing* the lost or corrupted parts of an image. The inpainting techniques are a fruitful sub-field in the area of digital image processings [21–23], which have been widely used in practice. As we will show in Section 3.4.1, inpainting is more effective at eliminating the heavy perturbations created by L_0 attacks than previous defenses.

We implement a system AEPecker to demonstrate the advantages aforementioned and the weakness of L_0 attacks. The system architecture is shown in Figure 3.1. After inputting an image I to a pre-processor $\mathcal{P}$, we obtain another image I'. Then, the Siamese network predicts whether I is adversarial by taking $\langle I, I' \rangle$ as the input pair. If I is detected as an L_0 AE, then we regard I' as a rectified image and use it to replace I in subsequent image classification for the defense purpose.

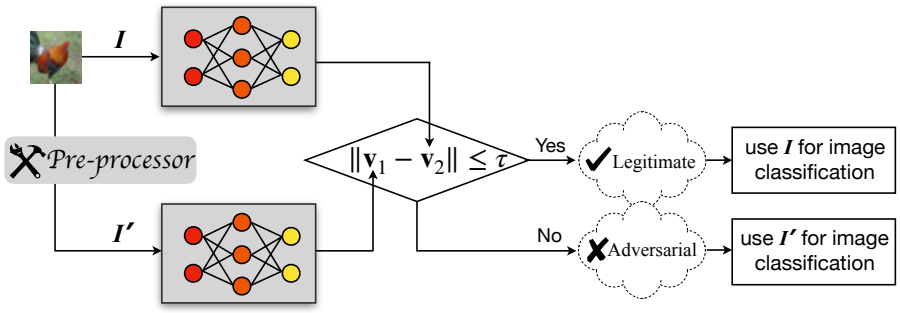


Figure 3.1: The architecture of AEPecker. [1]

We have evaluated our system in terms of its detection and defense capabilities using the popular image datasets CIFAR-10 and MNIST. Two leading L_0 AE generation methods, JSMA [7] and CW-L_0 [9], are both considered in the evaluation. In the case of CIFAR-10 (we have similar results for MNIST), the evaluation results show that (1) the detection rate on the CW-L_0 and JSMA attack is 97.1% and 99.7% respectively, both with a low false positive rate; (2) the proposed system has outstanding *transferability*, as a detector trained only with JSMA AEs can detect CW-L_0 AEs with a high detection rate (99.4%), and vice versa; (3) the detection is also *attack-target-model agnostic* (model agnostic, for short), since in the aforementioned experiments CW-L_0 AEs and JSMA AEs actually target different image classification models; and (4) our defense method recovers the classification accuracy from 0% (when classifying those successful AEs) to 87.3% for CW-L_0, and from 0% to 96.1% for JSMA, and meanwhile, has a very small impact on benign images.

Moreover, in order to illustrate the effectiveness of the Siamese network in detecting AEs, we experiment to use a preprocessing technique, *bit depth reduction*, that is known to be weak. Feature squeezing [17] used it as one of the pre-processors and obtained an AE detection rate 4.1%. In contrast, the Siamese network plus the weak preprocessing technique achieves 99.6%, which demonstrates the unique advantage of the Siamese architecture in detecting AEs.

[1] If I is detected as an L_0 AE, I' is used for image classification as a defense.

To summarize, in this chapter, we point out the inherent characteristics of L_0 AEs, which typically contain high-amplitude perturbations to very few and isolated pixels, and propose to exploit them to develop detection and defense techniques. In addition, we convert the L_0 *AE detection problem* into an *image comparison problem*, and propose to use a Siamese network to automatically extract the subtle discrepancies of the input pair as features for the AE detection. The detector demonstrates multiple prominent strengths, such as transferability across attacks and being attack-target-model agnostic (so the detector keeps effective across attack methods and target classifiers). More importantly, we propose an effective *inpainting*-based defense against L_0 perturbations, which can recover the correct classification at a high probability. To the best of our knowledge, this defense method achieves the highest accuracy when dealing with L_0 AEs. Last but not least, adaptive attacks that try to bypass our detection are considered and evaluated. The evaluation results show that our system is resilient to them.

3.2 System Design

The proposed system consists of a Siamese network (Section 3.2.2) which determines whether an input image is an L_0 AE, and a pre-processor (Section 3.2.1) which also can be used as a defense component to correct the classification under the existence of L_0 AEs. Note that the pre-processor has a very small impact on benign images; thus it can be used as a defense component independently without relying on detection.

3.2.1 Pre-processor

The pre-processor adopted in our system is designed to reduce adversarial noises while preserving the features in images to reduce false positives. From this perspective, the proposed pre-processor can also be deployed as a defense against L_0 attacks.

Intuitively, failing to limit the amplitude of those altered pixels in the images will result in outlier pixels. Previous work [54] emphasizes that it is challenging to get rid of the effect of those heavy perturbations. However, we argue the outlier pixels can be fixed by applying a processor based on *inpainting*. In image processing, the term "inpainting" refers to the process of reconstructing lost or corrupted regions of image data (or to remove small defects). Our idea is to treat those outliers as small corrupted regions, and the inpainting technique exactly meets the need for eliminating the L_0 noise.

In detail, we observe that those L_0 perturbations manifest themselves visually as salient noises. A mask to determine which pixels should be reconstructed can help identify these cases. When inspecting the pixel intensity in different color channels (e.g., the R, G, B channels for color images), for an altered pixel, it is highly possible that one *extreme value* can be observed in at least one channel. For example, an original pixel is represented as an intensity vector [0.32, 0.56, 0.62], where all the values are normalized. After corrupting by the L_0 attack, it becomes [0.33, 0.55, 0.96], whose B channel has an *extreme value* 0.96. We define a value as *extreme* if it is either smaller than an upper bound α or larger than an lower bound β. Thus, to obtain such a mask, we first locate all pixels of which the intensity are exceptional at least one channel. Meanwhile, we noticed that such pixels that achieve *extreme values* in all of the three channels are often the bright parts such as the sky in a natural image. Therefore, we use a parameter γ to help filter out such pixels in color images. According to our observation, we choose $\gamma = 0.7$ as an empirical value. Lines 4-10 in Algorithm 1 show the procedures to initially create the mask.

In addition, considering that the number of altered pixels only occupies a small portion of the image, the possibility that most of the altered pixels will assemble to form a connected region is very low. Consequently, to further exclude those unlikely candidates, we will remove those relatively large connected regions from the mask.

Algorithm 1: The Pre-processor based on Inpainting.

Input: A color image $\mathcal{I}$;
two bounds α and β used to find extreme values;
a parameter γ to describe *bright pixels* in natural images;
a *structuring element* $\mathcal{E}$ of the specified size and shape.

Output: A processed color image, denoted by $\mathcal{S}$.

1 Normalize $\mathcal{I} \leftarrow [min(\mathcal{I}) - \mathcal{I}]/[min(\mathcal{I}) - max(\mathcal{I})]$;
2 Extract three channels ($\mathcal{I}^R$, $\mathcal{I}^G$, $\mathcal{I}^B$) from $\mathcal{I}$;
3 Initialize the masks $\mathcal{M}^R, \mathcal{M}^G, \mathcal{M}^B \leftarrow \{0\}$;
4 **for** each pixel $\mathcal{I}_i \in \mathcal{I}$, **do**
5 **if** $(\mathcal{I}_i^R < \alpha) \vee [(\mathcal{I}_i^R > \beta) \wedge (\mathcal{I}_i^G \leq \gamma \ \vee \ \mathcal{I}_i^B \leq \gamma)]$ **then**
6 $\mathcal{M}_i^R \leftarrow 1$;
7 **if** $(\mathcal{I}_i^G < \alpha) \vee [(\mathcal{I}_i^G > \beta) \wedge (\mathcal{I}_i^R \leq \gamma \ \vee \ \mathcal{I}_i^B \leq \gamma)]$ **then**
8 $\mathcal{M}_i^G \leftarrow 1$;
9 **if** $(\mathcal{I}_i^B < \alpha) \vee [(\mathcal{I}_i^B > \beta) \wedge (\mathcal{I}_i^G \leq \gamma \ \vee \ \mathcal{I}_i^R \leq \gamma)]$ **then**
10 $\mathcal{M}_i^B \leftarrow 1$;
11 **for** each pixel $\mathcal{M}_i^\chi \in \mathcal{M}^\chi$, where $\chi := R, G, B$, **do**
12 **if** $\exists\ N(\mathcal{M}_i^\chi) > \mathcal{E}$, s.t. $\mathcal{M}_j^\chi = 1 \ \wedge \ \mathcal{M}_j^\chi \in N(\mathcal{M}_i^\chi)$ **then**
13 $\mathcal{M}_j^\chi \leftarrow 0$;
14 **for** each $\mathcal{I}^\chi := \mathcal{I}^R, \mathcal{I}^G, \mathcal{I}^B$, **do**
15 $\mathcal{S}^\chi \leftarrow$Inpainting $\mathcal{I}^\chi$ according to $\mathcal{M}^\chi$;
16 Reconstruct $\mathcal{S}$ with $\mathcal{S}^R$, $\mathcal{S}^G$ and $\mathcal{S}^B$;
17 **return** $\mathcal{S}$.

Specifically, we use a *structuring element* $\mathcal{E}$ to describe a connected region with the specified size and shape. If a connected region is larger than $\mathcal{E}$, we will exclude such region from the mask, as Lines 11-13 in Algorithm 1 show, where $N(\cdot)$ denotes a connected neighborhood.

We thus independently produce an inpainting mask for each channel of a color image. We then take advantage of the inpainting method proposed in [22] to restore those deteriorated pixels for each channel, as Lines 14-15 in Algorithm 1 show. Figure 3.2 displays some concrete examples applying Algorithm 1 with $\alpha = 0.2$, $\beta = 0.8$ on CIFAR-10. The resultant images in the even numbered row show that the adversarial perturbations are almost completely eliminated. We will provide more detailed

experimental results to demonstrate how our defense influences the effectiveness of L_0 attacks in Section 3.4.

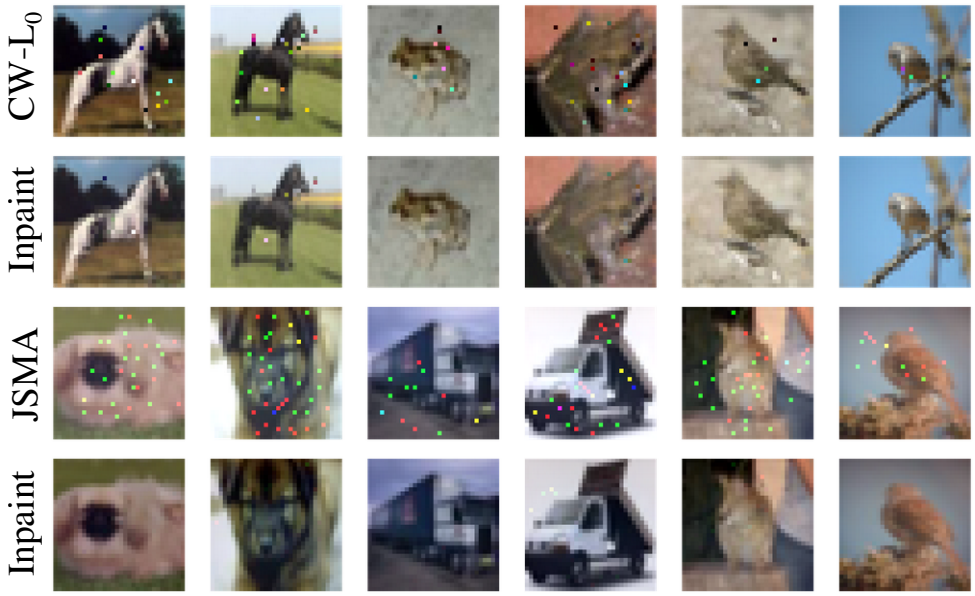


Figure 3.2: Defense based on inpainting.[2]

The algorithm for gray images is very similar to Algorithm 1, but we only need to consider one channel rather than three. Thus, we can consider the algorithm for gray images as a special case of the algorithm for color images.

Parameters selection. At beginning, our algorithm normalizes the value of all input pixels, such that their values are in the range of $[0, 1]$. (1) α is the upper bound of extremely small values; thus, the value of α should be small (e.g. less than 0.2). (2) β is the lower bound of extremely large values; thus, it should be relatively large (e.g. 0.7 at least). Different parameters settings slightly affect the effectiveness of rectifying AEs. We show the experiment results in Section 3.4.1. (3) In addition, as aforementioned, we use a parameter, γ, to help filter out the normal bright parts in a natural image. The term *atmospheric light* refers to those pixels, which has been discussed in detailed in the field of image processing [58]. Based on our experience, in

[2]The first and third rows show the CW-L_0 and JSMA attack applied to CIFAR-10 images, respectively. The second and fourth rows show the corresponding resulting images after inpainting.

our experiment (Section 3.4), the value of γ is set to 0.7. (4) Finally, the structuring element $\mathcal{E}$ is closely related to the restoration capability of the inpainting algorithm and the size of input images. The corrupted region that can be restored by the widely used inpainting algorithms is not only a single pixel but also a small patch [21–23]. However, as the size of patch increases, the restoration effect usually degrades. A recommendation size of $\mathcal{E}$ given by [22] ranges from three to ten pixels. Note that the performance of the pre-processor has little impact on the detection accuracy of AEPecker, as demonstrated in our evaluation (see Section 3.4.2.3).

3.2.2 Siamese Network-Based Detector

As a classic category of neural network architecture, Siamese networks [57] are widely applied among those tasks that involve detecting similarities or other relationships between two or more comparable things [59]. In general, a Siamese network consists of two sub-networks which share one identical architecture with the same weights.

Given an input image I, when pre-processing is adopted, the input image I and the pre-processed one I' may be very different even if I is benign. On the other hand, the discrepancy between *the two images*, I and I', may not be simply described using a single value and compared with a threshold, as adopted by *feature squeezing* [17]. These are the main challenges in devising an accurate detection technique.

We propose a Siamese-based L_0 AE detector with the help of a *pre-processor*, which converts the AE detection problem into an image comparison problem. Once the model with fine tuned weights is established (via training), the discrepancy between I and I' can be extracted by the Siamese network. Taking the discrepancies as features, the model can predict whether the input image is adversarial or not.

Figure 3.3 illustrates the architecture of our Siamese network-based AE detector. In particular, we learn from the classical AlexNet [60] to design our CNN-based sub-networks but only use a shallow network. The purpose is to explore how well the AE

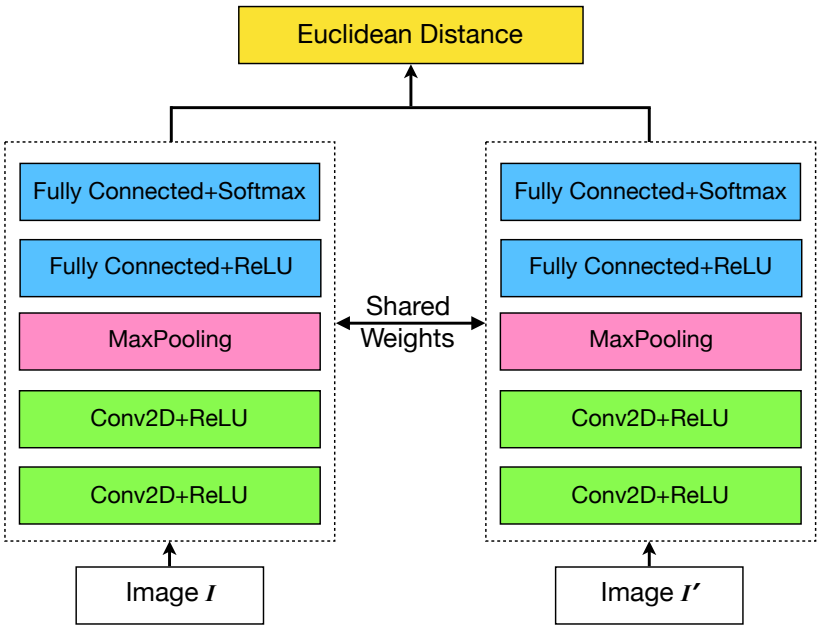


Figure 3.3: The architecture of a Siamese network which is used as our AE detector.

detector performs even when it only uses a simple network design. The details of the sub-network employed by each twin in the Siamese network are as follows:

$$
\begin{aligned}
CNN: \; & \rightarrow conv(3 \times 3, 64) \rightarrow ReLU \\
& \rightarrow conv(3 \times 3, 64) \rightarrow ReLU \\
& \rightarrow maxpool(2 \times 2) \rightarrow dropout(0.3) \\
& \rightarrow Flatten \\
& \rightarrow linear(\_\_\_, 128) \rightarrow ReLU \rightarrow dropout(0.5) \\
& \rightarrow linear(128, 10) \rightarrow softmax.
\end{aligned}
$$

Figure 3.4 elaborates the training phrase of the proposed detector based on a Siamese network. Given an image I and its pre-processed version I', the Siamese network takes $\langle I, I' \rangle$ as inputs, where the label is 0 if I is not an AE (denoted as I_o), or 1 if I is an AE (denoted as I_a). Although it is difficult to use a formula to describe the discrepancy between the input pair $\langle I_a, I'_a \rangle$ and the consistency between $\langle I_o, I'_o \rangle$, the Siamese network is effective in learning such relationship. Moreover, the consistency and discrepancy can be learned even when a non-powerful pre-processor is adopted,

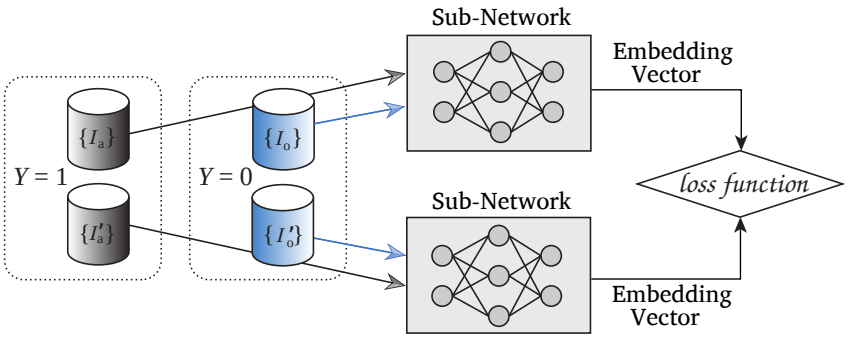


Figure 3.4: Training phrase of the AE detector based on a Siamese network.

such as a bit depth reducer (see Section 3.4.2.3). The result of the last layer of each of the two sub-networks is fed to a contrastive loss function [61]:

$$(1-Y)\frac{1}{2}(D_W)^2 + (Y)\frac{1}{2}\{(\max(0, m - D_W)\}^2$$

where D_W is defined as the Euclidean distance between the outputs of the two sub-networks, Y is a binary label assigned to the input pair, and $m > 0$ is a margin used to define a radius around the output of one of the sub-networks. Finally, once the model is successfully trained, the Siamese network can be used to determine whether I is an AE.

Our evaluation shows that, even with a relatively small training dataset and a network with very few layers, our detector can still achieve a very high accuracy.

3.3 Experimental Setup

In this section, we describe the experimental settings and implementation (Section 3.3.1) and discuss the datasets used in our evaluation (Section 3.3.2).

3.3.1 Experimental Settings

Threat model. We assume attackers have full knowledge on a trained target image classification model, but no ability to influence that model. Thus, given a trained tar-

get model, attackers can use the L_0 attacks including JSMA and CW-L_0 to generate AEs that will be misclassified by the target model.

Target models. We use two popular datasets for the image classification task: MNIST and CIFAR-10. For each dataset, we build up two individual models for the two types of L_0 attacks. Specifically, for MNIST, we set up a CNN-based classifier [62] for JSMA, and reuse the model structure provided in [9]—which we denote as Carlini$_M$—for CW-L_0. For CIFAR-10, we select the 32-layered ResNet model based on a residual learning framework [26] for JSMA, and reuse the model structure given in [9]—which we denote as Carlini$_C$—for CW-L_0. All the target models are trained from scratch.

Table 3.1: Classification accuracy of the target models.

Dataset	**Target Model**	**Accuracy**
MNIST	Carlini$_M$ [9]	99.26%
	CNN [62]	99.52%
CIFAR–10	Carlini$_C$ [9]	78.86%
	ResNet [26]	91.96%

Table 3.1 summarizes the classification accuracy on the testing data of each model. The accuracy of Carlini$_M$ and the CNN target model for MNIST is 99.26% and 99.52%, respectively; and the accuracy of Carlini$_C$ and the ResNet model for CIFAR-10 is 78.86% and 91.96%, respectively. Note that only those images which can *be correctly classified* by the corresponding target models are used to generate AEs in the following experiments.

Attacks. For the target models Carlini$_M$ and Carlini$_C$, we reuse the code provided in [9] to generate CW-L_0 AEs. The default parameters settings suggested by Carlini and Wagner [9] are as follows: the number of maximum iterations is 1000, the initial constant is 0.001, and the largest constant is 2^6. To compare with the state-of-the-art works [17, 24], we follow these parameters settings. Furthermore, for the target CNN and ResNet model, we generate AEs with JSMA by leveraging the Adversarial

Robustness Toolbox (ART) [13]. We used the same parameters settings as [17, 24], i.e., $\theta = 1, \gamma = 0.1$. As both JSMA and CW-L_0 are targeted attacks, we designate the *next* class as the target class.

Table 3.2: Evaluation of the L_0 attacks.

Dataset	**Attack**	**Success rate**
MNIST	CW-L_0 [9]	100%
	JSMA [7]	81.6%
CIFAR–10	CW-L_0 [9]	100%
	JSMA [7]	99.8%

Table 3.2 reports the results of the AEs. The *success rate* is defined as the probability that an adversary achieves their goal. For a targeted attack, it is only considered a success if the model predicts the target class. Note that we only use the AEs that can *successfully attack the target models* to evaluate the performance of our system on detecting AEs.

Implementation. We implement our Siamese-based detector in Python using the Keras [27] platform with TensorFlow [63] as backend. Keras provides a large number of high-level neural network APIs and can run on top of TensorFlow. The Telea's inpainting algorithm [22] is implemented based on Open Source Computer Vision Library (OpenCV) [64].

The experiments were performed on a computer running the Ubuntu 18.04 operating system with a 64-bit 3.6 GHz Intel® Core(TM) i7 CPU, 16 GB RAM and GeForce GTX 1070 GPU.

3.3.2 Data Preparation

We generate AEs based on two image datasets, i.e., CIFAR-10 and MNIST.

CIFAR-10 contains 60,000 color images; each is assigned to one of ten different classes, such as dog, frog and ship. CIFAR-10 is split into the training and testing dataset, which contains 50,000 and 10,000 images, respectively.

We first filter out those images that cannot be correctly classified by the corresponding target model. We then use the CW-L_0 algorithm to generate AEs that can *successfully attack* the Carlini$_C$ model [9], and create two dis-joint datasets, denoted as $\mathcal{D}_{\mathcal{C}}$-`CWL0-Train` and $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test`. In detail, $\mathcal{D}_{\mathcal{C}}$-`CWL0-Train` contains 10,000 legitimate images and 10,000 AEs. $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` contains 1,000 benign images and 1,000 AEs. Next, we follow the similar method on CIFAR-10 but instead using JSMA to generate AEs based on ResNet classifier [26]. As a result, we obtain two dis-joint datasets, denoted as $\mathcal{D}_{\mathcal{C}}$-`JSMA-Train` and $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test`. There are 10,000 legitimate images and 10,000 AEs in training set. There are 1,000 legitimate images and 1,000 AEs in testing set.

MNIST contains 70,000 8-bit grayscale images of hand-written digits. Each image is assigned a label from 0 to 9. MNIST is split into the training and testing dataset, which contains 60,000 and 10,000 images, respectively. We carry out similar procedures on MNIST to create a training and testing set but using different target models. As a result, we have $\mathcal{D}_{\mathcal{M}}$-`CWL0-Train` and $\mathcal{D}_{\mathcal{M}}$-`CWL0-Test` based on Carlini$_M$ model [9], as well as $\mathcal{D}_{\mathcal{M}}$-`JSMA-Train` and $\mathcal{D}_{\mathcal{M}}$-`JSMA-Test` based on CNN [62] model. The sizes of these datasets are the same as their counterparts in CIFAR-10. Considering that CIFAR-10 is a more challenging dataset compared with MNIST, we will spend more space on explaining the results for CIFAR-10 in the following experiments.

Note that all the aforementioned legitimate images can be classified correctly by the target model, and all the AEs can successfully fool the corresponding target model.

3.4 Evaluation

In this section, we first evaluate the effect of our pre-processing method as a defense alone (Section 3.4.1). Next we evaluate the accuracy of our system on detecting AEs generated by JSMA and CW-L_0 (Section 3.4.2), and the efficiency in terms of training

and testing (Section 3.4.3). The resilience of our system against an adaptive attack is presented in Section 3.5.

3.4.1 Effectiveness of Pre-processor as Defense

It is worth noting that our proposed method can not only detect adversarial examples but also rectify the classification results. Thus, this sub-section shows that our pre-processor as a defense can *individually* and functionally rectify the classification results of L_0 AEs.

To mislead a classifier to predict a specific target class, the adversarial perturbations produced by an L_0 attack such as JSMA or CW-L_0 are introduced intentionally instead of randomly. Moreover, the adversarial strength of an L_0 attack limits the number of pixels that can be manipulated; and as a result, the manipulated pixels need to have significant changes. The proposed inpainting-based pre-processor is to eliminate the possible adversarial pixels while preserving the benign ones. Therefore, the inpainting-based pre-processor can also be considered as a defense against L_0 attacks.

Inpainting-based pre-processor for color images. We first evaluate the effectiveness of the inpainting-based pre-processor as a defense against L_0 attack on CIFAR-10. The inpainting-based algorithm has two parameters: the threshold α extracts the pixels whose values tend to be very small, and β is used to screen all pixels whose values tend to be extremely large. We use 1,000 AEs in $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` to evaluate the effectiveness of the inpainting-based pre-processor as a defense against CW-L_0 attacks and examine its performance with varying values of α and β. Without pre-processing, these AEs result in 0% classification accuracy when using the Carlini$_C$ model [9]. After pre-processing, each recovered AE is analyzed by the model to predict a class label. Table 3.3 shows the results. When $\alpha = 0.1$ and $\beta = 0.7$, the

performance is the best—the classification accuracy on these AEs is increased from 0% to 87.3%.

Table 3.3: The classification accuracy on AEs in $\mathcal{D}_{\mathcal{C}}$-CWL0 -Test after using inpainting-based pre-processors.

β \ α	0.0	0.1	0.2
0.6	81.3%	86.9%	84.2%
0.7	80.5%	87.3%	86.5%
0.8	76.0%	86.2%	86.5%

We then use 1,000 AEs in $\mathcal{D}_{\mathcal{C}}$-JSMA-Test to evaluate the effectiveness of the inpainting-based pre-processor as a defense against JSMA attack. Without pre-processing, these AEs result in 0% classification accuracy of the ResNet model [26]. After applying inpainting-based pre-processor to rectify those AEs, each recovered AE is analyzed by the ResNet model to predict a class label. Table 3.4 shows the results. We can see that when $\alpha = 0.0$ and $\beta = 0.8$, the performance is the best—the classification accuracy on these AEs is increased from 0% to 96.1%. This classification accuracy is higher than 87.3% given by the previous Carlini$_C$ model. Moreover, the ResNet model is more robust against the benign perturbations introduced by the inpainting procedure.

Table 3.4: The classification accuracy on AEs in $\mathcal{D}_{\mathcal{C}}$-JSMA -Test after using inpainting-based pre-processors.

β \ α	0.0	0.1	0.2
0.6	90.0%	81.2%	63.2%
0.7	94.1%	88.8%	74.5%
0.8	96.1%	91.2%	77.3%

As a comparison, we examine the impact of both SVD compression and median filter on AEs generated by L_0 attacks. First, as a low-pass filter, SVD compression is usually used to reduce noise in images. As shown in Table 3.5, when varying the loss ratio of SVD compression, the classification accuracy on the processed AEs is

very low—at most 44.5% and 27.4% for CW-L_0 AEs and JSMA AEs, respectively. Note that we only use those images which can be correctly classified by the target model to generate AEs; thus the maximum classification accuracy given by the target model here is 100%. Therefore, the experiment suggests that the perceptible perturbations introduced by an L_0 attack are very difficult to be reduced when only using the frequency domain filters. Alternatively, [17] and [49] claim that median filter is particularly effective in mitigating adversarial examples generated by an L_0 attack because such perturbations are very similar to salt-and-pepper noises. In our experiments, after applying the median filter to process AEs in $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` and $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test`, the classification accuracy given by Carlini$_C$ and ResNet is 79.8% and 85.3%, respectively; both are lower than the proposed defense.

Table 3.5: The classification accuracy on testing datasets after applying SVD compression.

Loss ratio	60%	40%	20%
JSMA	27.4%	17.9%	5.4%
CW-L_0	44.5%	35.1%	21.4%

Inpainting-based pre-processor for gray images. We can observe similar results on MNIST when taking advantage of the inpainting-based pre-processor as a defense against an L_0 attack. Our experiment shows that processing the 1,000 AEs in $\mathcal{D}_{\mathcal{M}}$-`CWL0-Test` with the proposed inpainting-based method results in a significant increase of the classification accuracy on the Carlini$_M$ model [9]—from 0% to 88.2%. Similarly, after using the proposed inpainting-based method on the 1,000 AEs in $\mathcal{D}_{\mathcal{M}}$-`JSMA-Test`, the recovered AEs result in the classification accuracy on the CNN model [62] to increase from 0% to 86.1%. All of the results above are obtained when $\alpha = 0.1$ and $\beta = 0.8$. When varying the value of α from 0.1 to 0.2 and the value of β from 0.7 to 0.8, the classification accuracy increases slowly (84.9% at least).

As a comparison, Bafna et al. [65] independently focus on the L_0 attacks, and propose a defense based on Fourier transform. But our approach outperforms theirs—after applying their defense algorithm against CW-L_0, their classification accuracy on the MNIST testing set is only 72.8%. Note that they did not conduct experiments on standard color-image datasets such as CIFAR-10.

Impact on benign images. To investigate the impact of the defense methods on benign images, we first carry out an experiment on the 1,000 benign images from $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test`. Specifically, we use the ResNet model [26] to classify each color image after the inpainting-based defense is applied. The classification accuracy on these processed images only decreases from 100% to 95.6%. Next, we conduct a similar experiment on the 1,000 benign images from $\mathcal{D}_{\mathcal{M}}$-`JSMA-Test`. We use the CNN model [62] to classify each gray image after the inpainting-based defense is applied. As a result, the classification accuracy on these processed images only decreases from 100% to 99.7%. The results show that a very small impact is imposed on classifying benign images.

Summary. Therefore, the proposed inpainting-based algorithm is effective in defending against L_0 attacks such as CW-L_0 and JSMA. Moreover, our defense methods have a very small impact on benign images, which implies it can be directly applied without relying on detection.

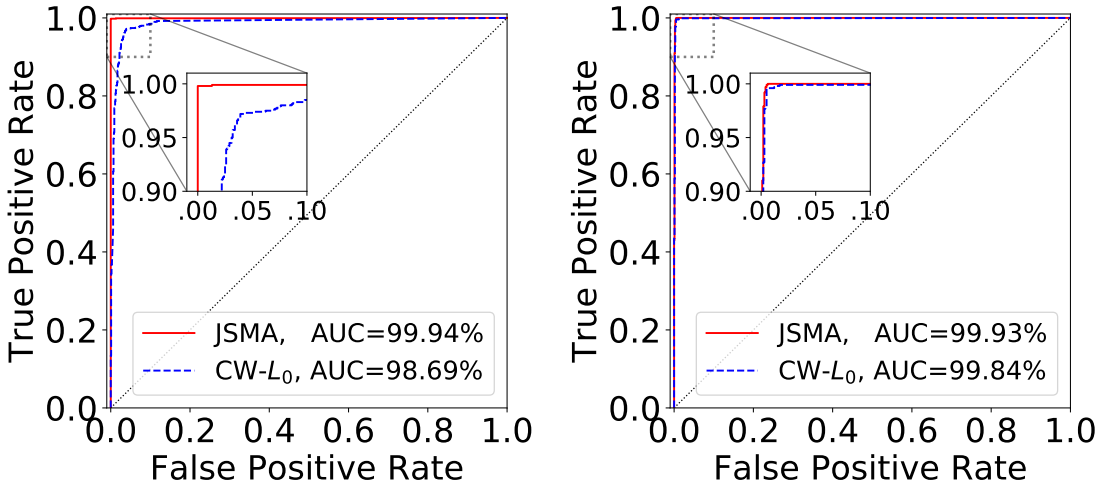
3.4.2 Detecting L_0 Adversarial Inputs

In this sub-section, we evaluate the effectiveness of our system on detecting AEs generated by L_0 attacks, which demonstrates that the detector (i.e., pre-processor plus the Siamese architecture) can effectively distinguish AEs from benign images.

3.4.2.1 Detection Efficacy

We evaluate the detection performance of the proposed scheme against CW-L_0 and JSMA attack. The inpainting-based pre-processor is used to create input pairs to the Siamese network.

Color images. The two training datasets, $\mathcal{D}_{\mathcal{C}}$-`CWL0-Train` and $\mathcal{D}_{\mathcal{C}}$-`JSMA-Train`, are used to train our system individually for 200 epochs using early stopping configured with a minimum accuracy change of 0.001 and 50 patience steps. If an accuracy change is less than 0.001, we consider that there is no improvement of the model performance; after 50 epochs with no improvement, the training is stopped. We save the resulting models as the base models.



(a) CIFAR-10 (b) MNIST

Figure 3.5: ROC curves for different datasets.

We now evaluate the detection accuracy of the base models against CW-L_0 and JSMA attack on $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` and $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test`, respectively. Each dataset contains 1,000 benign images and 1,000 AEs. We plot the ROC (receiver operating characteristic) curves, which are showed in Figure 3.5(a). We can achieve the AUC values of 98.69% and 99.94% for the two L_0 attacks. Table 3.6 shows more detailed results evaluated on the testing set.

We consider the adversarial images as positive, and the benign images as negative. Thus, the recall value (i.e., the detection rate) is the ratio of the number of successfully detected AEs to the total number of AEs; and False Positive Rate (FPR) is the fraction of the negative testing data (i.e., benign images) that is misclassified as positive. In practice, the distribution of adversarial and benign images are not balanced—most of the images should be benign. Thus, FPR is a very important metric to evaluate the model performance; a lower FPR indicates that the system makes fewer mistakes for benign images.

Table 3.6: The detection performance of the proposed system.

Dataset	**Attack**	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**FPR**
CIFAR-10	JSMA	99.85%	100.0%	99.70%	99.85%	0.0%
	CW-L_0	95.80%	94.64%	97.10%	95.85%	5.5%
MNIST	JSMA	99.80%	99.90%	99.70%	99.80%	0.1%
	CW-L_0	99.40%	99.30%	99.50%	99.40%	0.7%

As shown in Table 3.6, when analyzing AEs generated by the CW-L_0 attack, the detection rate of $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` is 97.1% and the FPR is 5.5%. When analyzing AEs generated by the JSMA attack, the detection rate of $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test` is 99.7% and the FPR is as low as 0.0%.

Gray images. We follow the same configurations to conduct an experiment on MNIST. Because gray images only have one channel, training a Siamese network on MNIST is simpler than that on CIFAR-10. We thus only train the detector for 100 epochs using an early stopping with 30 patience steps.

We evaluate the detection accuracy of the base models against CW-L_0 and JSMA attacks on $\mathcal{D}_{\mathcal{M}}$-`CWL0-Test` and $\mathcal{D}_{\mathcal{M}}$-`JSMA-Test`, respectively. We plot the ROC curves as the Figure 3.5 (b) shows. The AUC value can achieve 99.84% and 99.93%. In Table 3.6, we can observe similar results as the experiments given on CIFAR-10. When facing CW-L_0 attacks, the detection rate for the AEs from $\mathcal{D}_{\mathcal{M}}$-`CWL0-Test` is

99.5%, and the FPR is 0.7%. When facing JSMA attacks, the detection rate for the AEs from $\mathcal{D}_{\mathcal{M}}$-JSMA-Test can achieve 99.7%, and the FPR is as low as 0.1%.

Table 3.7: Comparison with state-of-the-art detectors in terms of FPR and detection rate.

Dataset	Detector	FPR	CW-L_0	JSMA
CIFAR-10	AEPecker	2.0%	98.4%	99.5%
	FS [17]	4.9%	98.1%	83.7%
	NIC [24]	3.8%	98.0%	94.0%
MNIST	AEPecker	0.4%	99.1%	99.3%
	FS [17]	4.0%	91.1%	100%
	NIC [24]	3.7%	100%	100%

Comparison. We compare the proposed system with the state-of-the-art AE detectors, including feature squeezing [17] and NIC [24]; both of them show that their systems are able to effectively detect L_0 AEs. Moreover, feature squeezing [17] uses multiple feature squeezers, and we only compare our system with the *best* results of their work. To this end, we train two comprehensive models for color images and gray images, respectively. Specifically, for color images, we train the detector using both $\mathcal{D}_{\mathcal{C}}$-CWL0-Train and $\mathcal{D}_{\mathcal{C}}$-JSMA-Train. For gray images, we train another detector using $\mathcal{D}_{\mathcal{M}}$-CWL0-Train and $\mathcal{D}_{\mathcal{M}}$-JSMA-Train. We summarize the adversarial detection rate and FPR in Table 3.7. For CIFAR-10, the detection rate of feature squeezing [17] on CW-L_0 and JSMA attacks is 98.1% and 83.7%, respectively, and its FPR—the percentage of the benign images among all the testing benign images that is misclassified as positive—is 4.9%. NIC [24] can achieve the detection rate of 98.0% and 94.0% on CW-L_0 and JSMA attacks respectively, and its FPR is 3.8%. Our AEPecker outperforms theirs—we can achieve the detection rate of 98.4% and 99.5% for the two types of L_0 attacks and our FPR is only 2.0%. With repect to MNIST, the detection rate of our model is comparable with feature squeezing [17] and NIC [24]. Moreover, AEPecker achieves the lowest FPR for both CIFAR-10

and MNIST. Therefore, our proposed detector outperforms the two state-of-the-art detectors.

3.4.2.2 Transferability

This experiment is to evaluate the transferability of our system: whether our system trained on one type of L_0 AEs can be directly applied to detect another type of L_0 AEs that have not been seen during training without any adaptation. To this end, we train our system using $\mathcal{D}_{\mathcal{C}}$-`JSMA-Train` and use $\mathcal{D}_{\mathcal{C}}$-`CWL0-Test` to test the detector. The result shows that the detection rate is as high as 99.4%. Similarly, we train our system using $\mathcal{D}_{\mathcal{C}}$-`CWL0-Train` and use $\mathcal{D}_{\mathcal{C}}$-`JSMA-Test` to test the detector. The result shows that the detection rate is as high as 98.7%.

The similar results are obtained for MNIST: if we use $\mathcal{D}_{\mathcal{M}}$-`JSMA-Train` to train the Siamese network and use $\mathcal{D}_{\mathcal{M}}$-`CWL0-Test` to test the detector, the detection rate is 96.3%; if our system is tained using $\mathcal{D}_{\mathcal{M}}$-`CWL0-Train` and tested on $\mathcal{D}_{\mathcal{M}}$-`JSMA-Test`, the detection rate can achieve 95.4%.

Summary. Therefore, our system has good transferability; our system trained on AEs generated by one L_0 attack can be directly applied to detect AEs generated by another L_0 attack without any adaptation.

3.4.2.3 Pre-processor Study

We next conduct an experiment to examine the impact of the pre-processor; specifically, we would like to see what the detection accuracy will be if a weak pre-processor is adopted. The *weak* here means the manipulated AEs through such a pre-processor still cannot be classified correctly by the target model with a high possibility. Through this, we will show that even with a weak pre-process, our system can still achieve a high detection accuracy—this means that a perfect pre-processor is unnecessary for our Siamese-based detector to achieve a high success rate of detection.

Without loss of generality, we use color images as an example for the following discussion. For color images such as CIFAR-10, each channel of RGB is encoded by 8 bits. As Figure 3.6 shows, we can reduce the original 8 bits to fewer bits without influencing the image recognizability for human eyes. Figure 3.6 also shows that it is very difficult to remove those striking adversarial perturbations introduced by L_0 attacks only with such an approach. Moreover, the original L_0 AEs can mislead the target neural networks to a classification accuracy of 0%. After applying bit depth reduction, the classification accuracy for AEs in the testing datasets is calculated. The experiment results are shown in Table 3.8, which suggest that processing the AEs generated by JSMA and CW-L_0 with bit depth reduction cannot increase the classification accuracy of the target model. Therefore, the bit depth reduction approach only has a very limited capability to defend against L_0 attacks.

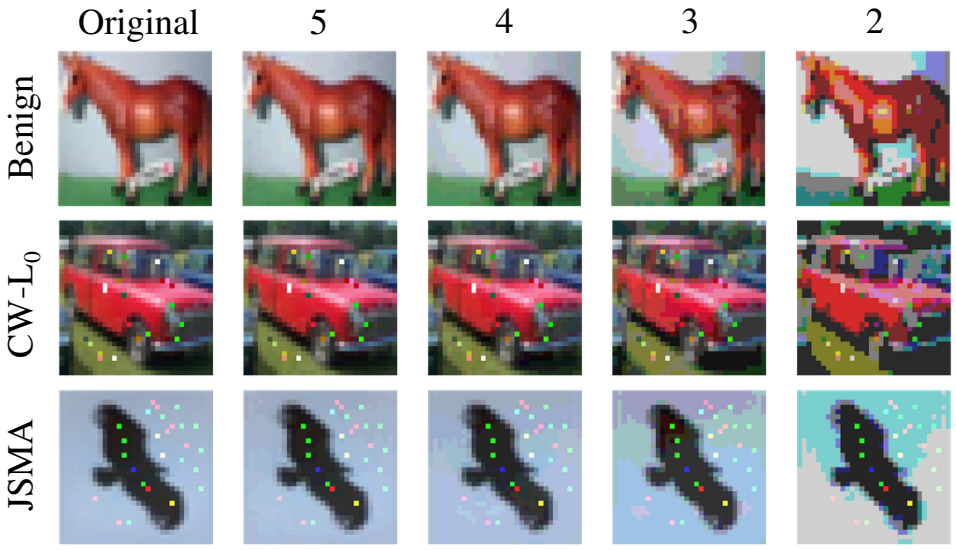


Figure 3.6: Image examples from CIFAR-10 after applying bit depth reduction.[3]

We thus choose the bit depth reduction as a weak pre-processor for color images. The experiment results show that even when a weak pre-processor (such as bit depth

[3]Given the different numbers of bit depth, the first row displays a benign image and its processed versions; the second row displays an AE generated by CW-L_0 and its corresponding processed images; the third row displays an AE generated by JSMA and its corresponding processed images.

reduction) is applied, our Siamese-based detector still can achieve a very good performance. The detection rates for AEs generated by JSMA and CW-L_0 are 99.6% and 99.4%, respectively; and the FPR is 2.1%. Xu et al. also use bit depth reduction as a pre-processor [17]; however, the best detection rates provided by their system for AEs generated by JSMA and CW-L_0 are 4.1% and 36.5% respectively, and its FPR is 5%.

Table 3.8: The classification accuracy for AEs in testing datasets after applying bit depth reduction.

Datasets	**Bit Depth**			
	2-bit	3-bit	4-bit	5-bit
$\mathcal{D}_{\mathcal{C}}$-JSMA-Test	22.2%	27.1%	21.2%	12.0%
$\mathcal{D}_{\mathcal{C}}$-CWL0-Test	51.2%	56.6%	55.1%	51.5%

Summary. Therefore, our proposed Siamese-based detector outperforms the state-of-the-art method when using the same weak pre-processor. The result also demonstrates that the good performance of our detector does not rely on a perfect pre-processor, but is due to the Siamese network design.

3.4.3 Efficiency

Training time. It is widely known that neural networks usually require a large amount of data and time for training. However, as our sub-networks employed within the Siamese architecture are quite simple and shallow, the training is very efficient. For example, for $\mathcal{D}_{\mathcal{M}}$-JSMA-Train and $\mathcal{D}_{\mathcal{C}}$-JSMA-Train, each epoch with 20,000 images (10,000 benign images and 10,000 AEs) only takes 5 and 7 seconds, respectively. On the other hand, due to the simple and shallow sub-networks, with a relatively small training set, our Siamese neural-network-based AEPecker can still achieve high detection accuracies (Section 3.4.2). Moreover, the training time is linear with respect to the number of epochs and the number of training samples for each epoch.

Our experiment results show that our system trained on CIFAR-10 and MNIST can converge very quickly and achieve high accuracy within 100 and 200 epochs, respectively—thus, the training only requires around 8 minutes and 23 minutes, respectively.

Testing time. The trained detector can detect an AE very fast. For example, AEPecker only takes approximately 0.5ms on average to detect whether an image from CIFAR-10 is adversarial or not.

3.5 Resilience to Adaptive Attack

An adversary who knows the details of AEPecker will try to adapt the attacks. Thus, we seek to understand the resilience of AEPecker to adaptive attacks by answering the following questions. (**Q1**) What is the percentage of the high-amplitude altered pixels in AEs generated using non-adaptive L_0 attacks? Exploration of this question not only helps us understand L_0 attacks and why the proposed detection and defense techniques work well, but also guides the adversary to adapt L_0 attacks. (**Q2**) How difficult is it for an adversary to adaptively generate L_0 AEs that bypass our detection?

To answer these questions, we launch an adaptive L_0 attack by adopting a similar method described in [18], which has successfully demonstrated a capability to impede *feature squeezing* [17]. Our implementation is based on the source code given by [9]. To generate L_0 AEs, after each step of stochastic gradient descent (SGD), an intermediate distorted image is generated as a resolution of the optimizer. Each time the optimizer runs, the process tries to minimizes the number of altered pixels and, in the meanwhile, keep the targeted attack successful.

Answer to Q1. Let $N_{\mathcal{A}}$ be the number of altered pixels and $N_{\mathcal{E}}$ the number of such altered pixels that possess *extreme values* (i.e., values smaller than α or larger than β). We consider the ratio $\rho = N_{\mathcal{E}}/N_{\mathcal{A}}$ as an indicator showing the percentage

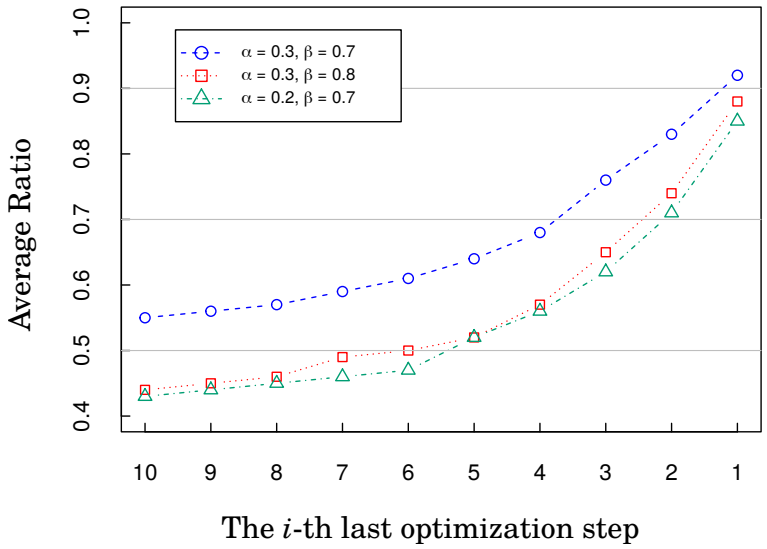


Figure 3.7: L_0 attacks are launched on 100 randomly selected images from CIFAR-10. For each of the last 10 optimization steps, we examine the average ratio $\bar{\rho}$ of the 100 intermediate distorted images.

of pixels with large-amplitude perturbations, and want to understand how this ratio changes in the AE generation process. As an empirical analysis, we carry out SGD step by step on 100 randomly selected images from CIFAR-10. For each of the last 10 steps, we calculate an average ratio $\bar{\rho}$ value of the 100 intermediate images, as shown in Figure 3.7. We observe that the average ratio $\bar{\rho}$ goes higher and higher as $N_{\mathcal{A}}$ decreases. Finally, when the optimal resolution is found, around 90% of the altered pixels by average possess extreme values. This helps understand why the proposed technique works well, since it is designed to deal with such large-amplitude perturbations by recovering these pixels.

An adversary who is aware of the details of the proposed technique thus should try to control the amplitude of those altered pixels while satisfying the L_0 optimization target (i.e., minimizing $N_{\mathcal{A}}$). Thus, given an image, we run the SGD multiple times; once the value of ρ is over 80% (note this value finally can reach 90% by average), we explore different optimization paths. The result shows that only 5% of the cases

succeed to control the ratio ρ under 80%. Therefore, it is difficult to control the amplitude of the altered pixels while satisfying the L_0 optimization target.

Answer to Q2. We follow the procedures described in [18] to adaptively search potential L_0 AEs. Our design of the adaptive L_0 attack is as follows. Since inpainting is used in both detection and defense, the adversary integrates it into the AE generation; during the AE generation, the intermediate image at each step of the optimization procedure is processed using our inpainting pre-processor. Next, we check whether the resulting image is a successful attack. If that it cannot successfully fool the neural network, we iteratively run SGD multiple times (10 in our experiments) until a resolution is found. We randomly select 100 color images from CIFAR-10, Our experiments show that the final number of altered pixels only takes up less than 2% of the total number of the pixels in images from CIFAR-10, which means that they achieve the L_0 optimization target. In order to save computational time, we start checking and adaptive optimization after such percentage is lower than 5%. The result finally shows that only 7% of cases can generate successful AEs to evade our detection. In contrast, [18] shows that adaptive attacks using a similar method can bypass *feature squeezing* [17] at 100%. Therefore, our method is much more resilient than prior work.

Summary. Based on these explorations, we conclude that L_0 attacks have an inherent limitation, and it is difficult for adaptive attacks to overcome the limitation to bypass our detection.

3.6 Related Work

As an increasing number of adversarial example generation approaches have been proposed in recent years, AE attacks are becoming a non-negligible threat to the deep learning. Consequently, finding solutions which may protect DNNs against AE attacks is of significance. Generally, the protection strategies against AE attacks fall

into two groups, i.e., detection and defense. In this section, we will briefly review them both.

3.6.1 Detecting Adversarial Examples

An AE detector is a binary classifier which is designed to distinguish an adversarial sample from a legitimate one. There are two strategies which are often used to design AE detectors, i.e., adversarial training and predication mismatch.

Detector Training. Some techniques use both AEs and legitimate images to train a detector. For example, Li et al. [14] extract PCA features after inner convolutional layers of the neural network, then use a cascade classifier to detect AEs. Metzen et al. [15] use both adversarial and benign samples to train a CNN-based auxiliary network. This light-weight sub-network works with the target model to detect AEs. They usually require a large number of samples to train the model while we only need a relatively small dataset. More importantly, our detector achieves a high detection rate but low FPR for handling L_0 AEs. In addition, Ruan et al. [66] provided an empirical analysis and conclude that the model which is trained only through using both adversarial and benign data usually shows poor performance because the normal-related features and adversarial-related features are overlapped.

Prediction Mismatch. Feeding one benign image to different DNNs often gains very similar prediction, while using one adversarial image as input, different models may make distinct predictions. Based on this observation, the disagreement among several models can be leveraged as a measurement to distinguish whether the input sample is benign or adversarial. For example, Bagnall et al. [55] train an ensemble of multiple models to use a rank voting mechanism to combine those outputs. In this way, an ensemble disagreement can be used to detect adversarial examples. *Bi-model* [53] firstly employs two pre-trained distinct models to generate features, then feeds the concatenated features to an additional binary classifier.

Other works [16, 17, 37, 54] apply pre-processors on an input image. For example, Meng et al. [16] train an auto-encoder as the image filter. Tian et al. [37] pre-process the images with randomly rotation and shifting since adversarial examples are usually sensitive to such transformation operations. Liang et al. [54] implement an adaptive denoiser based on image entropy as the filter. Their methods then feed the original image and the processed one to the same neural network–if the predictions of the two images fail to match, the input is adversarial. In addition, Xu et al. [17] propose *feature squeezing* to detect AEs by comparing the prediction on original inputs with that on the squeezed ones. However, the proposed detector outperforms *feature squeezing* for handling L_0 attacks. Note that the performance of their approach heavily relies on the effectiveness of the feature squeezing methods. On the contrary, our Siamese-based detector does not rely on powerful pre-processing (see Section 3.4.2.3).

Our detection technique seems close to the approaches in the second category but is actually very different. Rather than using a simple mismatch or a distance value to describe the discrepancy between an AE and its manipulated image, our technique uses a Siamese network to automatically extract the discrepancy between the two as features for detection.

3.6.2 Defense

The primary task of defensive techniques is to alleviate or eliminate the influences of AEs. In other words, with the help of defense, even facing attacks, those deep learning models are still able to make correct predictions at a high possibility. In general, the current defensive techniques can be grouped into two major categories, that is, model enhancement and input transformation.

Model Enhancement. The first category improves the resilience of neural networks by including AEs in the process of model training, i.e., *adversarial training* [6, 67]. However, this type of defense is usually less effective against black-box attacks than

white-box attacks considering the training only focuses on one certain neural network. Also, Xu et al. claim that this kind of technique suffers high cost because of iterative re-training with both adversarial and normal examples [17]. Alternatively, *defensive distillation* is proposed, and can obstruct the neural networks from fitting too tightly to the data [11]. However, the prior work [9] demonstrates that the approach can be easily circumvented with a minimal modified attack such as a CW-L_0. *Shield* [46] enhances a model by re-training it with multiple levels of compressed images based upon JPEG. However, this method is still ineffective against L_0 attacks.

Input Transformation. For the second category of defenses, researchers have averted their eyes from neural networks to the adversarial inputs themselves. In short, pre-processing the inputs before feeding them to networks is helpful for increasing the prediction accuracy even when facing adversarial examples. The reason why adversarial examples could successfully fool the deep learning model without being perceived is that attackers take advantage of the information redundancy of images to add adversarial noise. Consequently, well designed filters or denoisers can be considered a cure for adversarial images by removing unwanted noise. For instance, Liao et al. [47] propose higher-level guided denoisers to remove the adversarial noise from inputs; however, their approach is computationally expensive and their work does not show its effectiveness on L_0 attacks. Some other methods adopt compression techniques, such as PCA [68] and JPEG [48, 49, 69, 70], to filter out the information redundancy which may provide living space for adversarial perturbations in images; however, these approaches are not suitable for L_0 attacks. Furthermore, Bafna et al. [65] independently propose a defense against L_0 attacks; but their Fourier-transform-based approach is not as effective as ours (see Section 3.4.1).

There exist some approaches that do not fall in either category. For example, MVP-Ears [52] borrows the idea of multi-version programming from software engineering and applies it to audio AE detection. It deploys multiple diverse automatic

speech recognition systems in parallel, and detects audio AEs by comparing their recognition results. However, the idea will probably fail in handling image AEs, which are known to have good transferability [6].

3.7 Summary

In the setting of classic L_0 AE attacks, a bounded number of pixels can be corrupted without limiting the amplitude. These large-amplitude perturbations in L_0 AEs are considered as a challenge by many previous works, since the effect of such corruptions is difficult to eliminate. Considering the threats caused by L_0 AEs, a highly accurate detection technique and an effective defense that can rectify the classification under L_0 perturbations are urgently needed. By identifying and exploiting the inherent characteristics of L_0 AEs, we develop AEPecker that thwarts this type of attacks. Its novel Siamese-network-based design shows very high accuracies in detecting L_0 AEs, and its inpainting-based preprocessing technique can effectively rectify those AEs and thus correct the classification results. Plus, it is resilient to adaptive attacks that bypass prior approaches.

Chapter 4

Comprehensive Adversarial Examples Detector with Hybrid Design

4.1 Introduction

In the previous chapters, we have demonstrated the noticeable characteristics of L_0 and L_2 attacks, so we tailor our detectors accordingly. Next, we will systematically combine two proposed detecting techniques together to cover all categories of AEs, i.e., L_0, L_2, and L_∞ attacks.

To integrate the L_∞ AEs into existing detection frameworks, we statistically compare different kinds of AEs in terms of the distortion degree. In fact, researchers have investigated the overall perturbations caused by various AE attacks and found that L_2 often lead to lower distortion by taking both the number of altered pixels and their amplitude into consideration. For example, Liang et al. pointed out that "CW L_2 attack can find adversarial examples with at least two times lower distortion than FGSM" [54]. Furthermore, Table 4.1 and Table 4.2 (which are extracted from Table 2 in [17]) quantitatively show the results evaluated on 100 seed images. In particular, "the L_0 distortion is normalized by the number of pixels (e.g., 0.560 means 56% of all pixels in the image are modified)" [17]. We can see that the amplitude of altered pixels caused by L_0 is remarkable. The difference between the number of altered pixels by L_2 and L_∞ is not statistically significant. Even from the perspective of the L_∞ distance, the difference is not significantly higher. But, the overall distortions

Table 4.1: Distortion evaluation of attacks on CIFAR-10.

Type	**Attack**	**Success Rate**	**Distortion**		
			L_∞	L_2	L_0
L_∞	FGSM	85%	0.016	0.863	0.997
	IGSM	92%	0.008	0.368	0.993
	CW-L_∞	100%	0.012	0.446	0.990
L_2	DeepFool	98%	0.028	0.235	0.995
	CW-L_2	100%	0.034	0.288	0.768
L_0	JSMA	98%	0.896	4.954	0.079
	CW-L_0	100%	0.650	2.103	0.019

Table 4.2: Distortion evaluation of attacks on ImageNet.

Type	**Attack**	**Success Rate**	**Distortion**		
			L_∞	L_2	L_0
L_∞	FGSM	99%	0.008	3.009	0.994
	IGSM	100%	0.004	1.406	0.984
	CW-L_∞	99%	0.006	1.312	0.850
L_2	DeepFool	89%	0.027	0.726	0.984
	CW-L_2	90%	0.019	0.666	0.323
L_0	CW-L_0	100%	0.898	6.825	0.003

in terms of Euclidean distance by L_2 is approximately two times to four times lower than the L_∞ AEs.

Therefore, we use a uniform Siamese-network-based detector like AEPecker to tackle both L_0 and L_∞ AEs. If the images cannot be labelled as adversarial in this step, they will be further checked by the follow-up Themis. We will detailedly describe our hybrid system design in Section 4.2.

This proposed comprehensive AE detector acquires the strengths from the two aforementioned detection techniques. In this way, the new detector not only inherits all noticeable features from it assembly components, but also is extended to cover all kinds of AEs. Our evaluation shows that it is able to accurately distinguish various AEs with a low false positive rate on benign images.

4.2 System Design

The Figure 4.1 shows the system architecture of our proposed hybrid AE detector. It consists of two modules, i.e. AEPecker and Themis. After feeding an image I and its counterpart I' (that is manipulated by the pre-processor) into AEPecker, this module can decide whether I is adversarial. If not, I will be input to Themis, so that a further check can be made. Only if the latter module outputs a negative label again, can I be confirmed as benign.

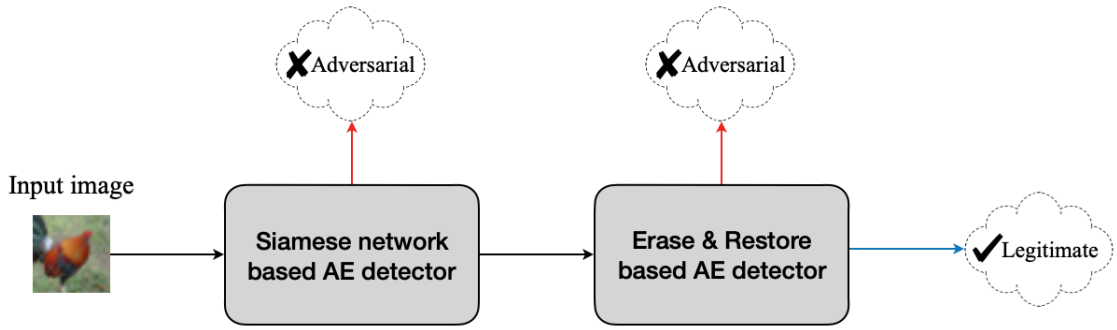


Figure 4.1: The architecture of proposed detector.

4.2.1 Pre-processor

In the original design of AEPecker, only our proposed inpaiting-based pre-processor is deployed. To enhance the performance of our hybrid detector, we propose using k-means based color quantization as a color depth reducer after applying the inpaiting-based pre-processor. Therefore, in our new system, we manipulate an input image I using the proposed inpaiting-based method first. After that, we process the resulting image with k-means based color quantization to obtain I'.

One pixel in a true-color image has three color channels, i.e. R, G and B, each of which also needs an 8-bit integer to represent a component in such channel. Consequently, a color image theoretically provides 2^{24} possible values for each pixel. In reality, we can significantly reduce the the number of color in an original image without fundamentally decreasing the image recognizability to humans.

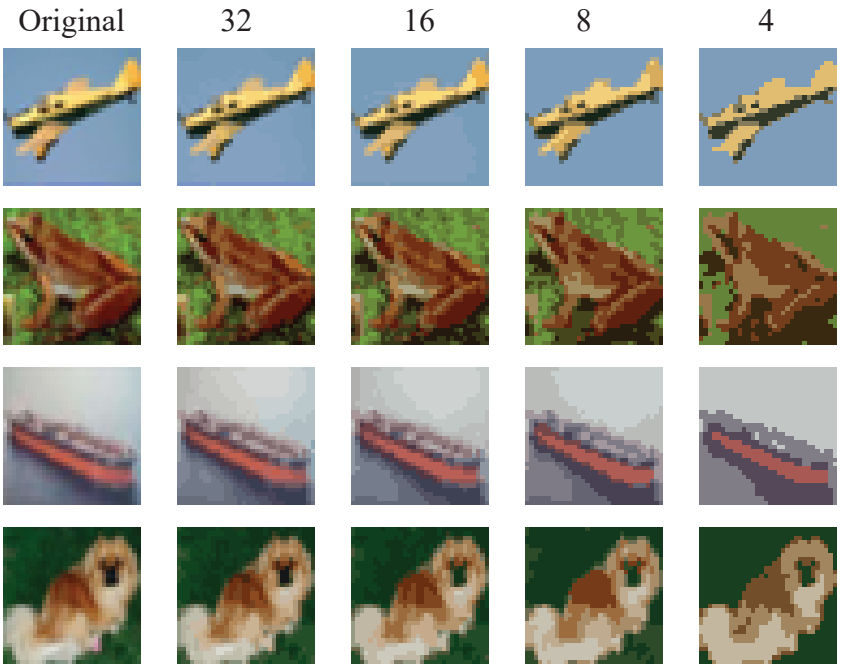


Figure 4.2: Images with color quantization based on the k-means algorithm. [1]

Almost any clustering algorithm in 3D space can be applied as a color quantizater. In particular, Celebi [71] analyzed the performance of k-means as a color quantizer and demonstrated it can outperform other color quantization methods. Therefore, we perform a k-means based color quantization for a given image with N unique colors. In the resultant image, N is finally reduced to k, where $k \ll N$. Figure 4.2 shows some concrete instances where the original color images are from CIFAR-10 dataset. With k-means based color quantizer, we obtained the corresponding resultant images with different numbers of unique colors.

The value selection for the parameter k depends on the number of unique colors in the image and the size of image. We investigate the impacts of different values of k on the adversarial perturbation reduction, taking CIFAR-10 as an example. Only successful AEs are used in our evaluation, so without k-means based color quantization, the target DNN cannot make a correct prediction, i.e. the classification

[1]The first column shows the original images from CIFAR-10 dataset. The following columns show the corresponding versions with different numbers of unique colors after being pre-processed.

Table 4.3: Impacts of different values of k (CIFAR-10).

Attack	k			
	4	8	16	32
FGSM	45.5%	49.8%	36.8%	23.6%
IGSM	43.5%	42.4%	18.9%	2.7%
JSMA	38.6%	34.8%	31.2%	27.9%

accuracies are all 0%. As Table 4.3 shows, after using the proposed pre-processor, the influence of adversarial perturbations are reduced. Considering the size of images from CIFAR-10 is small, as k increases, the effect of k-means based color quantization weakens. The pre-processor deployed in our system aims to reduce the influence of adversarial perturbation but keep the features in images which may help a target DNN to make correct predictions. Therefore, we empirically adopt $k = 8$ in the following experiments.

4.2.2 Model Training

Since the L_∞ AEs detection is integrated into the AEPecker framework, we re-train this Siamese-network-based module. In particular, Given an image I and its pre-processed version I', the Siamese network takes $\langle I, I' \rangle$ as inputs, where the label is positive if I is either an L_0 or L_∞ AE. Otherwise, if I is a benign image or an L_2 AE, we label the image pair $\langle I, I' \rangle$ as negative in the dataset.

It is worth noting that Themis works as a plugin to enhance the overall detection system, thus no extra changes are needed for it.

4.3 Evaluation

As summarized in Table 4.4, we compare the proposed hybrid detector with other prior works [17, 24, 25]. We use the same attack settings as used in these works. Our detector outperforms Feature Squeezing [17] and LID [25] in terms of detection rate

and FPR. Nic [24] achieves a roughly comparable performance with the proposed detector, but ours FPR of is still the lowest.

Table 4.4: Comparison with other prior detectors in terms of detection rate and FPR.

Detector	FPR	Attack					
		FGSM	IGSM	DeepFooL	CW-L_2	JSMA	CW-L_0
Proposed	3.3%	98%	98%	99%	100%	99%	98%
NIC	4.2%	100%	100%	91%	96%	94%	98%
FS	4.9%	21%	55%	77%	100%	84%	98%
LID	5.6%	94%	96%	84%	86%	92%	90%

4.4 Summary

Although the previously proposed Themis and AEPecker are not a panacea for detecting against all possible attacks, it easy to combine them together or deploy them as a complement to tackle a wider variety of AE attacks. To demonstrate this point, we propose a comprehensive AE detector acquires the strengths from its two assembly modules. In particular, we propose an enhanced pre-processor and re-train the Siamese-network-based detector. The evaluation results show that our system outperforms other state-of-the-art detectors in terms of both the high detection rate and the low FPR.

Chapter 5

Conclusion

The existence of adversarial examples is considered as a fatal threat to the neural-network-based applications. Therefore, how to effectively protect the neural- network against AE attacks attracts more and more interests from the research community. We have observed that many AE detectors have been proposed. However, recent studies show that the detection usually goes ineffective when facing adaptive L_2 AEs. To this end, we propose a novel detection technique against DNN adversarial examples, based on the observation that L_2 AEs are sensitive to the Erase-and-Restore operations, while benign samples are not. To provide a deeper understanding of the proposed technique, we have qualitatively and quantitatively interpreted the sensitivity distinction of AEs and benign images to E&R operations. The evaluation results show that the proposed detector outperforms other state-of-the-art approaches in terms of both detection rates and false positive rates. In addition, our detector is target-model agnostic, keeps effective across different L_2 attack methods (i.e., good transferability across attack methods), and is resilient to adaptive attacks. Furthermore, unlike many other detection methods that are complex and thus difficult to construct and train, this method is very simple to build and easy to apply in practice.

Another category of AEs which are considered as a challenge by many previous works is the L_0 AE. In the setting of classic L_0 AE attacks, a bounded number of pixels can be corrupted without limiting the amplitude. Many researchers point out that the effect of heavy perturbations caused by the L_0 attack is difficult to eliminate. However, we argue that this noticeable characteristic is an limitation of

L_0 attacks which leads such AEs can be exploitable. Thus, we develop AEPecker that thwarts this type of attacks. Its novel Siamese network based design shows very high accuracies in detecting L_0 AEs, and its inpainting-based preprocessing technique can effectively rectify those AEs and thus correct the classification results. Plus, it is resilient to adaptive attacks that bypass prior approaches.

Finally, to cover a wider variety of AE attacks, we combine the previously proposed methods together to obtain a comprehensive hybrid AE detector. This new detection system acquires both strengths from its two assembly modules. The evaluation results demonstrate that our system outperforms other state-of-the-art detectors in terms of both the high detection rate and the low FPR.

5.1 Discussion and Future Work

First, our work shows that only controlling the number of altered pixels without limiting the resulting amplitude weakens the power of the generated AEs. Thus, for the purpose of AE generation, how to make a good trade-off between the number of altered pixels and their amplitude becomes critical when designing new AE generation algorithms, which is a good direction worth being explored.

In addition, our adaptive attack (following the procedures in [18]) is based on the exploration of different optimization paths. There exist some other alternative white-box attacks, such as the method proposed in [20] which attempts to create new AEs by modifying the loss functions to bypass detectors. Whether other different adaptive attacks, such as [20], can bypass our detector is interesting, and we plan to investigate it in the future.

Another possible adaptive attack is to limit the perturbations in a restricted area that the defender is not aware of. Most prior works [72–74] that limit perturbed pixels to a given sub-region use L_0-norm. We notice that some recent works [75, 76] that only perturb pixels in a limited region also use L_2-norm to achieve better

invisibility. However, their modified regions or even pixels are predictable, which can be exploited by an AE detector. Therefore, how to limit the L_2 perturbation to an arbitrary sub-region is still an open question. A future task is to investigate the effectiveness of E&R once such L_2 perturbations are available.

Furthermore, this work focuses on attacks launched against digital images; we notice that physical attacks [77, 78] are attracting more and more interests from the research community. In particular, patch-based AEs, which are widely used in physical attacks, are not in the scope of this work. However, it is interesting to study the effectiveness of E&R on physical attacks [77]. We leave this as our future work.

Finally, some recent studies on certified robustness have attracted much interest from the research community. For example, Cohen et al. [79] present a certified robustness guarantee in norm for the smoothed classifier that is obtained by using Gaussian noise. Furthermore, Jia et al. [80] derive a tight robustness in norm for top-predictions when using randomized smoothing with Gaussian noise. Some related works [81, 82] also show that inpainting has a side effect of denoising by smoothing the interpolated pixels. Our inpainting-based approach can be considered as an alternative to randomized smoothing. Thus, it is interesting to analyze the certified accuracy of our inpainting-based method. We plan to explore this in our future work.

Bibliography

[1] S. Ren, K. He, R. Girshick, and J. Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015.

[2] N. Wang and D.-Y. Yeung. "Learning a deep compact image representation for visual tracking". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2013.

[3] O. M. Parkhi, A. Vedaldi, and A. Zisserman. "Deep face recognition". In: *British Machine Vision Conference (BMVC)*. 2015.

[4] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. "A discriminative feature learning approach for deep face recognition". In: *European Conference on Computer Vision (ECCV)*. 2016.

[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. "Intriguing properties of neural networks". In: *International Conference on Learning Representations (ICLR)*. 2014.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy. "Explaining and Harnessing Adversarial Examples". In: *International Conference on Learning Representations (ICLR)*. 2015.

[7] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. "The limitations of deep learning in adversarial settings". In: *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE. 2016, pp. 372–387.

[8] A. Kurakin, I. Goodfellow, and S. Bengio. “Adversarial examples in the physical world”. In: *ICLR workshop*. 2017.

[9] N. Carlini and D. Wagner. “Towards evaluating the robustness of neural networks”. In: *IEEE Symposium on Security and Privacy (SP)*. 2017.

[10] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. “DeepFool: a simple and accurate method to fool deep neural networks”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[11] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. “Distillation as a defense to adversarial perturbations against deep neural networks”. In: *IEEE Symposium on Security and Privacy (SP)*. 2016.

[12] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *International Conference on Learning Representations (ICLR)*. 2015.

[13] M.-I. Nicolae, M. Sinn, M. N. Tran, B. Buesser, A. Rawat, M. Wistuba, V. Zantedeschi, N. Baracaldo, B. Chen, H. Ludwig, I. Molloy, and B. Edwards. *Adversarial Robustness Toolbox v1.0.1*. https://adversarial-robustness-toolbox.readthedocs.io/en/latest/modules/attacks/evasion.html#carlini-and-wagner-l-2-attack. 2018.

[14] X. Li and F. Li. “Adversarial examples detection in deep networks with convolutional filter statistics”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[15] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. “On detecting adversarial perturbations”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[16] D. Meng and H. Chen. “MagNet: a two-pronged defense against adversarial examples”. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2017.

[17] W. Xu, D. Evans, and Y. Qi. “Feature squeezing: Detecting adversarial examples in deep neural networks”. In: *Network and Distributed System Security Symposium (NDSS)*. 2018.

[18] W. He, J. Wei, X. Chen, N. Carlini, and D. Song. “Adversarial example defenses: Ensembles of weak defenses are not strong”. In: *11th USENIX Workshop on Offensive Technologies (WOOT)*. 2017.

[19] N. Carlini and D. Wagner. “Magnet and “efficient defenses against adversarial attacks" are not robust to adversarial examples”. In: *arXiv preprint: 1711.08478* (2017).

[20] N. Carlini and D. Wagner. “Adversarial examples are not easily detected: Bypassing ten detection methods”. In: *ACM Workshop on Artificial Intelligence and Security*. 2017.

[21] J. Shen and T. F. Chan. “Mathematical models for local nontexture inpaintings”. In: *SIAM Journal on Applied Mathematics* 62.3 (2002).

[22] A. Telea. “An image inpainting technique based on the fast marching method”. In: *Journal of Graphics Tools* 9.1 (2004).

[23] J. Mairal, M. Elad, and G. Sapiro. “Sparse representation for color image restoration”. In: *IEEE Transactions on image processing* 17.1 (2007).

[24] S. Ma, Y. Liu, G. Tao, W.-C. Lee, and X. Zhang. “NIC: Detecting Adversarial Samples with Neural Network Invariant Checking”. In: *Network and Distributed System Security Symposium (NDSS)*. 2019.

[25] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. “Characterizing adversarial subspaces using local intrinsic dimensionality”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[26] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[27] F. Chollet. *Keras*. `https://keras.io`. 2015.

[28] F. Zuo, B. Yang, X. Li, L. Luo, and Q. Zeng. "Exploiting the Inherent Limitation of L_0 Adversarial Examples". In: *International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. 2019.

[29] J. Rauber, W. Brendel, and M. Bethge. "Foolbox: A Python toolbox to benchmark the robustness of machine learning models". In: *arXiv preprint: 1707.04131* (2017).

[30] S. Diamond, V. Sitzmann, S. Boyd, G. Wetzstein, and F. Heide. "Dirty pixels: Optimizing image classification architectures for raw sensor data". In: *arXiv preprint: 1701.06487* (2017).

[31] G. B. P. da Costa, W. A. Contato, T. S. Nazare, J. E. Neto, and M. Ponti. "An empirical study on the effects of different types of noise in image classification tasks". In: *arXiv preprint: 1609.02781* (2016).

[32] S. Dodge and L. Karam. "Understanding how image quality affects deep neural networks". In: *IEEE International Conference on Quality of Multimedia Experience (QoMEX)*. 2016.

[33] Y. Freund and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of computer and system sciences* 55.1 (1997).

[34] C. Cortes and V. Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995).

[35] Z. Gong, W. Wang, and W.-S. Ku. "Adversarial and clean data are not twins". In: *arXiv preprint: 1704.04960* (2017).

[36] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. “On the (statistical) detection of adversarial examples”. In: *arXiv preprint: 1702.06280* (2017).

[37] S. Tian, G. Yang, and Y. Cai. “Detecting Adversarial Examples through Image Transformation”. In: *AAAI Conference on Artificial Intelligence*. 2018.

[38] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer. “Deflecting adversarial attacks with pixel deflection”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[39] A. Athalye, N. Carlini, and D. Wagner. “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples”. In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*. 2018.

[40] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. “Synthesizing Robust Adversarial Examples”. In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*. 2018.

[41] C. Villani. “The Wasserstein distances”. In: *Optimal Transport*. Springer, 2009, pp. 93–111.

[42] S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[43] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. “Improving the robustness of deep neural networks via stability training”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[44] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. “Towards deep learning models resistant to adversarial attacks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[45] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. “Thermometer encoding: One hot way to resist adversarial examples”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[46] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau. “Shield: Fast, practical defense and vaccination for deep learning using JPEG compression”. In: *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018.

[47] F. Liao, M. Liang, Y. Dong, T. Pang, J. Zhu, and X. Hu. “Defense against adversarial attacks using high-level representation guided denoiser”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[48] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer. “Protecting JPEG images against adversarial attacks”. In: *2018 Data Compression Conference*. IEEE. 2018, pp. 137–146.

[49] C. Guo, M. Rana, M. Cisse, and L. van der Maaten. “Countering adversarial images using input transformations”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[50] P. Gupta and E. Rahtu. “CIIDefence: Defeating Adversarial Attacks by Fusing Class-Specific Image Inpainting and Image Denoising”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2019.

[51] K.-C. Chen, P.-Y. Chen, and C.-M. Yu. “Poster: REMIX: Mitigating Adversarial Perturbation by Reforming, Masking and Inpainting”. In: *IEEE Symposium on Security and Privacy (SP)*. 2018.

[52] Q. Zeng, J. Su, C. Fu, G. Kayas, L. Luo, X. Du, C. C. Tan, and J. Wu. “A multiversion programming inspired approach to detecting audio adversarial examples”. In: *49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. 2019.

[53] J. Monteiro, Z. Akhtar, and T. H. Falk. “Generalizable Adversarial Examples Detection Based on Bi-model Decision Mismatch”. In: *arXiv preprint: 1802.07770* (2018).

[54] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang. “Detecting adversarial image examples in deep neural networks with adaptive noise reduction”. In: *IEEE Transactions on Dependable and Secure Computing* (2018).

[55] A. Bagnall, R. Bunescu, and G. Stewart. “Training ensembles to detect adversarial examples”. In: *arXiv preprint: 1712.04006* (2017).

[56] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille. “Mitigating adversarial effects through randomization”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[57] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. “Signature verification using a "Siamese" time delay neural network”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 1994.

[58] J.-H. Kim, W.-D. Jang, J.-Y. Sim, and C.-S. Kim. “Optimized contrast enhancement for real-time image and video dehazing”. In: *Journal of Visual Communication and Image Representation* 24.3 (2013).

[59] F. Zuo, X. Li, P. Young, L. Luo, Q. Zeng, and Z. Zhang. “Neural Machine Translation Inspired Binary Code Similarity Comparison beyond Function Pairs”. In: *Network and Distributed System Security Symposium (NDSS)*. 2019.

[60] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2012.

[61] R. Hadsell, S. Chopra, and Y. LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2006.

[62] Y. Katariya. *Applying Convolutional Neural Network on the MNIST dataset*. `https://github.com/yashk2810`. 2017.

[63] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, et al. "Tensorflow: a system for large-scale machine learning." In: *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 2016.

[64] G. Bradski et al. *OpenCV*. `https://opencv.org`. 2017.

[65] M. Bafna, J. Murtagh, and N. Vyas. "Thwarting Adversarial Examples: An L_0-Robust Sparse Fourier Transform". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018, pp. 10075–10085.

[66] Y. Ruan and J. Dai. "TwinNet: A Double Sub-Network Framework for Detecting Universal Adversarial Perturbations". In: *Future Internet* 10.3 (2018).

[67] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint: 1706.06083* (2017).

[68] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal. "Enhancing robustness of machine learning systems via data transformations". In: *IEEE Conference on Information Sciences and Systems*. 2018.

[69] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau. "Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression". In: *arXiv preprint: 1705.02900* (2017).

[70] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy. "A study of the effect of JPG compression on adversarial images". In: *arXiv preprint: 1608.00853* (2016).

[71] M. E. Celebi. "Improving the performance of k-means for color quantization". In: *Image and Vision Computing* 29.4 (2011).

[72] H. Kwon, H. Yoon, and D. Choi. "Restricted evasion attack: Generation of restricted-area adversarial example". In: *IEEE Access* 7 (2019), pp. 60908–60919.

[73] A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard. "Sparsefool: a few pixels make a big difference". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[74] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein. "Are adversarial examples inevitable?" In: *International Conference on Learning Representations (ICLR)*. 2019.

[75] T. Deng and Z. Zeng. "Generate adversarial examples by spatially perturbing on the meaningful area". In: *Pattern Recognition Letters* 125 (2019), pp. 632–638.

[76] X. Dong, D. Chen, J. Bao, C. Qin, L. Yuan, W. Zhang, N. Yu, and D. Chen. "GreedyFool: Distortion-Aware Sparse Adversarial Attack". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[77] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. "Robust physical-world attacks on deep learning visual classification". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[78] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, F. Tramer, A. Prakash, T. Kohno, and D. Song. "Physical adversarial examples for object detectors". In: *12th USENIX Workshop on Offensive Technologies (WOOT)*. 2018.

[79] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. "Certified adversarial robustness via randomized smoothing". In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. 2019.

[80] J. Jia, X. Cao, B. Wang, and N. Z. Gong. "Certified robustness for top-k predictions against adversarial perturbations via randomized smoothing". In: *International Conference on Learning Representations (ICLR)*. 2020.

[81] R. D. Adam, P. Peter, and J. Weickert. "Denoising by inpainting". In: *International Conference on Scale Space and Variational Methods in Computer Vision*. Springer. 2017, pp. 121–132.

[82] T. Sanders and C. Dwyer. "Inpainting versus denoising for dose reduction in scanning-beam microscopies". In: *IEEE Transactions on Image Processing* 29 (2019), pp. 351–359.

Appendix A

PUBLICATIONS

1. F. Zuo and Q. Zeng. "Exploiting the Sensitivity of L_2 Adversarial Examples to Erase-and-Restore." The 16th ACM Asia Conference on Computer and Communications Security (ASIACCS), 2021.

2. L. Luo, Q. Zeng, B. Yang, F. Zuo, and J. Wang. "Westworld: Fuzzing-Assisted Remote Dynamic Symbolic Execution of Smart Apps on IoT Cloud Platforms." The 37th Annual Computer Security Applications Conference (ACSAC), 2021.

3. F. Zuo, B. Yang, X. Li, L. Luo, and Q. Zeng. "Exploiting the Inherent Limitation of L_0 Adversarial Examples." The 22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID), 2019.

4. F. Zuo, X. Li, P. Young, L. Luo, Q. Zeng, and Z. Zhang. "Neural Machine Translation Inspired Binary Code Similarity Comparison beyond Function Pairs." The Network and Distributed System Security Symposium (NDSS), 2019

5. X. Li, F. Yan, F. Zuo, Q. Zeng, and L. Luo. "Touch Well Before Use: Intuitive and Secure Authentication for IoT Devices." The 25th Annual International Conference on Mobile Computing and Networking (MobiCom), 2019.

6. J. Qi, F. Zuo, H. Samet, and J.C. Yao. "K-Regret Queries Using Multiplicative Utility Functions." ACM Transactions on Database Systems (TODS), 43(2), pp.1-41. 2018.