Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 8319249, 17 pages
https://doi.org/10.1155/2020/8319249

Research Article

Complete Defense Framework to Protect Deep Neural Networks against Adversarial Examples

Guangling Sun,[1] Yuying Su,[1] Chuan Qin,[2] Wenbo Xu,[1] Xiaofeng Lu,[1] and Andrzej Ceglowski[3]

[1]*School of Communication and Information Engineering, Shanghai University, Shanghai 20044, China*
[2]*School of Optical-Electrical and Computer Engineering, University of Shanghai Science and Technology, Shanghai 200093, China*
[3]*Department of Accounting, Monash University, Caulfield East, Melbourne, VIC 3145, Australia*

Correspondence should be addressed to Xiaofeng Lu; luxiaofeng@shu.edu.cn

Received 11 January 2020; Revised 27 March 2020; Accepted 16 April 2020; Published 11 May 2020

Academic Editor: Bogdan Smolka

Although Deep Neural Networks (DNNs) have achieved great success on various applications, investigations have increasingly shown DNNs to be highly vulnerable when adversarial examples are used as input. Here, we present a comprehensive defense framework to protect DNNs against adversarial examples. First, we present statistical and minor alteration detectors to filter out adversarial examples contaminated by noticeable and unnoticeable perturbations, respectively. Then, we ensemble the detectors, a deep Residual Generative Network (ResGN), and an adversarially trained targeted network, to construct a complete defense framework. In this framework, the ResGN is our previously proposed network which is used to remove adversarial perturbations, and the adversarially trained targeted network is a network that is learned through adversarial training. Specifically, once the detectors determine an input example to be adversarial, it is cleaned by ResGN and then classified by the adversarially trained targeted network; otherwise, it is directly classified by this network. We empirically evaluate the proposed complete defense on ImageNet dataset. The results confirm the robustness against current representative attacking methods including fast gradient sign method, randomized fast gradient sign method, basic iterative method, universal adversarial perturbations, DeepFool method, and Carlini & Wagner method.

1. Introduction

Lately, the performance of deep neural networks (DNNs) on various applications, such as computer vision [1], natural language processing [2], and speech recognition [3], has been impressive. However, recent investigations also revealed that DNNs are fragile and are easily confused by adversarial examples contaminated by elaborately designed perturbations [4–7]. Szegedy et al. [4] first crafted some adversarial examples that are misclassified by DNNs with high probability. These adversarial examples easily deceived the targeted network even though they did not affect human recognition (see Figure 1). Undoubtedly, these adversarial examples are serious potential threats to security concerned applications such as autonomous vehicle systems [8] and face recognition [9]. Therefore, improving the robustness of DNNs to adversarial examples is of crucial importance.

To date, roughly three categories of approaches have been used to defend against adversarial examples. The first is to ensure that neural networks are robust against adversarial examples, the second is to reform adversarial examples, and the third is to detect adversarial examples. With respect to the first type, one of the most effective strategies is to (re) train the targeted network with adversarial examples to obtain an adversarially trained targeted network. However, each of these approaches has its respective limitations. The first approach cannot effectively defend against adversarial examples if they have not been learned while training the network. The second approach would inevitably have an

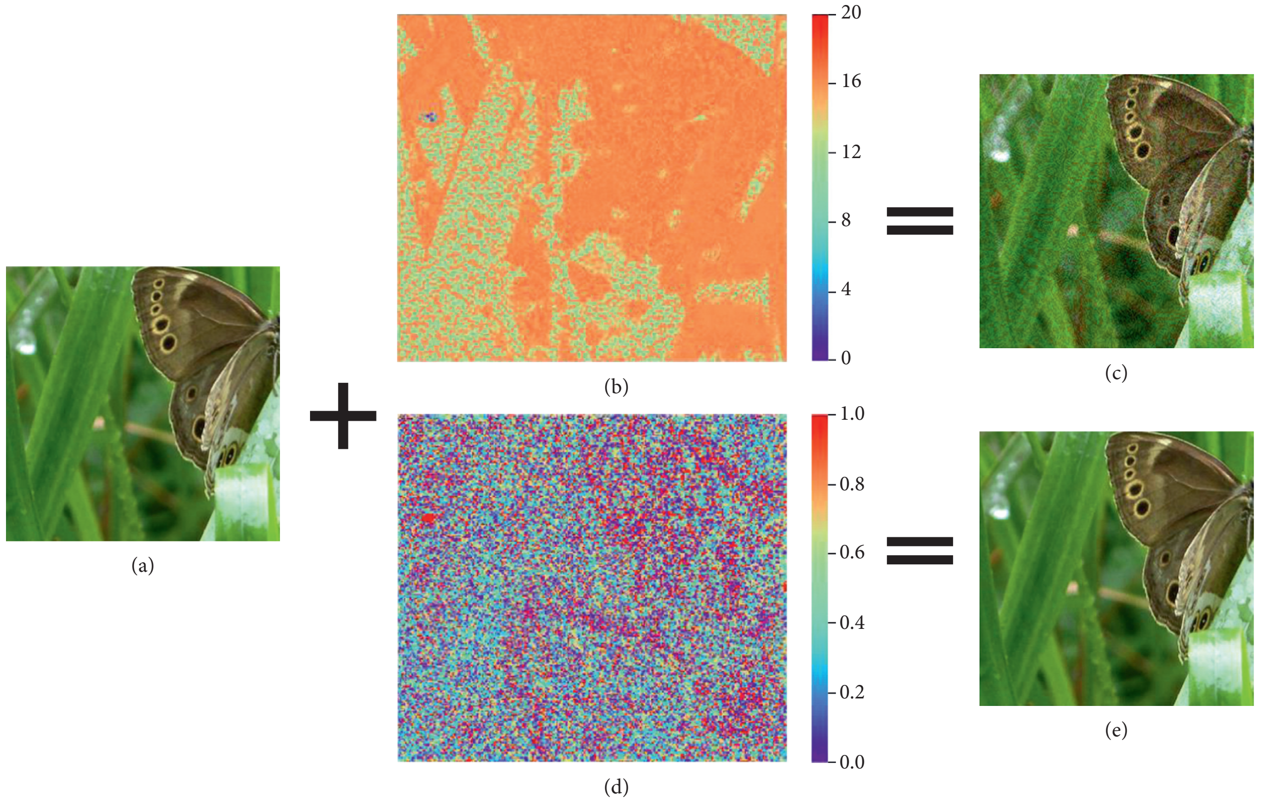

FIGURE 1: The illustration of an adversarial example with noticeable and unnoticeable perturbations. (a) The legitimate image is classified as "ringlet butterfly" with 98.1% confidence. (b) and (d) show the noticeable and unnoticeable adversarial perturbations, respectively. The corresponding adversarial images generated by fast gradient sign and Carlini & Wagner methods are misclassified as "starfish" with 97.6% confidence (c) and "chickadee" with 95.2% confidence (e) Note the different colors in (b) and (d) represent the average pixel values of three channels of the residual image which is the difference between adversarial and legitimate images. The color close to blue means the small difference and the color close to orange means the large difference so that they can be distinguished between noticeable and unnoticeable perturbations (the pixel values of legitimate and adversarial images range from 0 to 255).

impact on legitimate examples as they would also be reformed. The third approach may reject the detected adversarial examples, which might be unacceptable in certain scenarios.

To overcome the aforementioned drawbacks, we propose a complete defense framework that combines the three types of approaches. It consists of two detectors, a deep residual generative network (ResGN) [10], and an adversarially trained targeted network [11]. The two detectors detect adversarial examples; ResGN removes or mitigates adversarial perturbations once examples are detected as adversarial; and the adversarially trained targeted network classifies both the legitimate and cleaned examples (see Figure 2). The motivation for developing the comprehensive defense framework is that it meets the requirement of accepting all input examples and avoids altering legitimate examples as far as possible. This occurs simultaneously in that, once the examples are determined to be legitimate, they are not processed by the ResGN. Moreover, compared with a nonadversarially trained targeted network, an adversarially trained targeted network delivers superior performance on escaped adversarial examples and erroneously cleaned legitimate examples. Even for the cleaned adversarial examples, the latter network still outperforms the former since it is impossible for ResGN to perfectly remove the various perturbations.

On the contrary, we have discovered that the proposed attacks are implemented along the gradient-based and optimization-based attacks. Gradient-based attack just increases or decreases a loss function that depends on the gradients to seek an adversarial example, while optimization-based attack directly takes the minimal adversarial perturbation as one of the objective functions to optimize. Consequently, in general, the gradient-based attacks will introduce more visible flaws than the optimization-based attacks at the same attacking rate. According to their corresponding perturbations magnitudes, we call them noticeable perturbations and unnoticeable perturbations, respectively, in the remainder of the paper. In addition, to illustrate the distinguishment, we show the two types of perturbations in Figure 1. Adversarial examples with noticeable perturbations contain large statistical abnormalities that can be readily distinguished from legitimate ones, whereas unnoticeable perturbations can be readily destroyed. This led us to design a pair of complementary detectors: a statistical detector and a minor alteration detector. The statistical detector relies on extracting

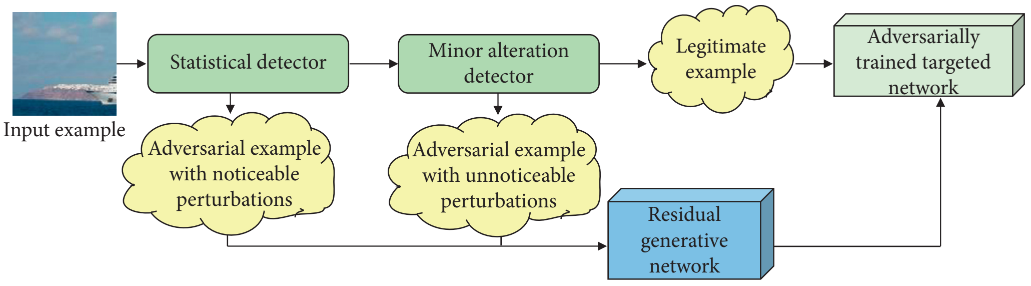
Input example
Statistical detector
Minor alteration detector
Legitimate example
Adversarially trained targeted network
Adversarial example with noticeable perturbations
Adversarial example with unnoticeable perturbations
Residual generative network

Figure 2: An overview of proposed complete defense framework. The statistical and minor alteration detectors filter out adversarial examples and the ResGN cleans the adversarial examples. Then, the cleaned examples and legitimate examples are classified by the adversarially trained targeted network.

statistical features from an input image to distinguish adversarial image from legitimate one. The minor alteration detector relies on crafting minor alterations to an input image and discovering the difference in output between the original and altered examples to distinguish adversarial example from legitimate one. This aspect differs in two respects from our previous work [12]: the first is that we extract a detection feature from three channels instead of using the average of three channels; the other is that minor geometric alterations are performed instead of Gaussian noise corruption on the input image. The improvement in the detection performance resulting from the two changes is verified by our empirical results.

We summarize our contributions as follows:

(1) We designed two detectors, a statistical and a minor alteration detector, which are adaptive to the characteristics of adversarial perturbations and filter out adversarial examples contaminated by noticeable and unnoticeable perturbations, respectively.

(2) We used an adversarially trained targeted network to classify examples. Thus, the two detectors, ResGN, and the adversarially trained targeted network form a complete defense framework.

(3) We conducted comprehensive experiments on the ImageNet dataset [13] to confirm the effectiveness of the proposed complete defense framework.

The paper is organized as follows. In Section 2, adversarial attacks and defensive techniques are briefly surveyed. In Section 3, the proposed complete defense framework is presented and analyzed. In Section 4, the comprehensive experiments are described to verify the effectiveness of the proposed complete defense framework. Section 5 draws the conclusions.

2. Background

Although attack is opposite to defense, attacking study is essential to increase the adversarial robustness of networks.

2.1. Attacks. In terms of the knowledge known by the adversary regarding the targeted model, attacks are grouped into white box, gray box, and black box attacks. In white box attacks, the adversary knows the structure and parameters of the targeted network, the training data, and even the defensive scheme of the defender. Of the three categories of attacks, white box attacks are the most frequently used when evaluating defensive techniques. In the following, we review widely used white box attacks.

2.1.1. Fast Gradient Sign Method (FGSM). Goodfellow [5] developed FGSM. It is a single-step attack that uses the ℓ_∞ metric measuring the distance between a legitimate and perturbed example. Formally, the adversarial example $\mathbf{x}^{\text{adv}}$ is obtained as follows:

$$\text{untargeted}: \ \mathbf{x}^{\text{adv}} = \mathbf{x} + \epsilon \cdot \operatorname{sign}\left(\nabla_{\mathbf{x}} J\left(g(\mathbf{x}), y_{\text{true}}\right)\right), \tag{1}$$

$$\text{targeted}: \ \mathbf{x}^{\text{adv}} = \mathbf{x} - \ \epsilon \cdot \operatorname{sign}\left(\nabla_{\mathbf{x}} J\left(g(\mathbf{x}), y_{\text{target}}\right)\right), \tag{2}$$

where $g(\cdot)$ is the classification result of the targeted network, $\nabla_{\mathbf{x}} J(\cdot,\cdot)$ is the gradient of the cost function $J(\cdot,\cdot)$ with respect to $\boldsymbol{x}$, and the value of ϵ controls the strength of the perturbation. Although FGSM is efficient, it introduces noticeable perturbations.

2.1.2. Randomized Fast Gradient Sign Method (R-FGSM). An improved version of FGSM, R-FGSM, was proposed by Tramèr [14]. It injects a small amount of random noise into the legitimate example before performing the FGSM attack. Specifically, for $\alpha < \epsilon$, a legitimate example x is corrupted into $\mathbf{x}^{\mathbf{R}}$ by the additive Gaussian noise:

$$\mathbf{x}^{R} = x + \alpha \cdot \operatorname{sign}\left(\mathcal{N}\left(0^{n}, \mathbf{I}^{n}\right)\right), \tag{3}$$

after which, the FGSM attack is performed on $\mathbf{x}^{R}$, as in equations (1) and (2).

2.1.3. Basic Iterative Method (BIM). Kurakin et al. [15] proposed the BIM attack, which is an iterative version of FGSM. During each iterative cycle, an intermediate result is yielded by the FGSM attack with a small step size of α. The intermediate result is clipped to ensure that the adversarial example remains in the ϵ-neighborhood of the legitimate example. For the ith iterative cycle, the adversarial example is generated as follows:

$$\begin{aligned}\mathbf{x}_0^{\text{adv}} &= \mathbf{x},\\ \text{untargeted}: \mathbf{x}_{i+1}^{\text{adv}} &= \text{clip}_{\epsilon,\mathbf{x}}\left(\mathbf{x}_i^{\text{adv}} + \alpha \cdot \text{sign}\left(\nabla_{\mathbf{x}} J\left(g\left(\mathbf{x}_i^{\text{adv}}\right), y_{\text{true}}\right)\right)\right),\\ \text{targeted}: \mathbf{x}_{i+1}^{\text{adv}} &= \text{clip}_{\epsilon,\mathbf{x}}\left(\mathbf{x}_i^{\text{adv}} - \alpha \cdot \text{sign}\left(\nabla_{\mathbf{x}} J\left(g\left(\mathbf{x}_i^{\text{adv}}\right), y_{\text{target}}\right)\right)\right).\end{aligned} \tag{4}$$

BIM has a much higher attacking rate than FGSM and it still causes noticeable perturbations even though fewer visual flaws occur than those crafted by FGSM.

2.1.4. DeepFool Method (DeepFool). Moosavi-Dezfooli et al. [16] proposed an iterative attack based on the ℓ_2 norm to compute the minimal distortion for a given example. The attacker assumes that the legitimate example resides in the region restricted by the decision boundaries of the classifier. This algorithm disturbs the example by a small vector. Then, the resulting example is taken to the boundary of the polyhedron, which is obtained by linearizing the boundaries of the region within which the image resides during each iterative step. The final perturbation is calculated by accumulating the perturbations added to the legitimate example in each iterative step, which forces the perturbed image to change its ground truth (GT) label. The formulation of DeepFool is as follows:

$$\mathbf{z}^* = \text{argmin}_{\mathbf{z}} \|\mathbf{z}\|_2 \quad \text{s.t. } g(\mathbf{x}+\mathbf{z}) \neq g(\mathbf{x}). \tag{5}$$

The perturbations introduced by DeepFool are unnoticeable and the attacking rate is much higher than that of FGSM and BIM.

2.1.5. Carlini & Wagner Method (CW). In an attempt to counter defensive distillation, Carlini and Wagner [17] introduced optimization-based adversarial attacks that render the perturbations quasi-imperceptible by restricting their ℓ_2, ℓ_∞, and ℓ_0 norms. The researchers demonstrated that distilled targeted networks almost completely fail against these attacks. In addition, the adversarial examples generated using a targeted network without distillation can be transferred successfully to a network with distillation. These facts indicate that the perturbations are suitable for black box attacks. In the remainder of this paper, CW_UT and CW_T denote untargeted and targeted CW attacks in ℓ_2 norms, respectively.

2.1.6. Universal Adversarial Perturbations (UAP). Moosavi-Dezfooli et al. [18] developed an attack generating universal adversarial perturbations that are image-agnostic. In their problem context, given the targeted classifier g and data distribution S, the existence of small universal perturbations ρ whose magnitude is measured by the ℓ_p norm with $p \in [1, \infty)$ and leading to most misclassified images is examined. The problem can be formulated as follows:

$$\begin{gathered} \text{P}(g(x) \neq g(x+\rho)) \geq 1-\delta \quad \text{s.t. } \|\rho\|_p \leq \epsilon,\\ x \sim S. \end{gathered} \tag{6}$$

The parameter ϵ controls the magnitude of the perturbation ρ and δ quantifies the failure rate of fooling the targeted network for all images sampled from distribution S. The UAP attack is implemented iteratively and the iteration will not terminate until most of the sampled images are misclassified by the targeted network.

2.2. Defensive Techniques. There is a rich literature relating to defensive strategies. Here, we outline the major defensive strategies into five sections.

2.2.1. Adversarial Training. By augmenting training samples with adversarial examples, adversarial training enhances the robustness of the network. Goodfellow et al. [5] and Huang et al. [19] evaluated their adversarial training only on the MNIST dataset and it is thought that adversarial training had provided regularization for the DNNs. Kurakin et al. [11] presented a comprehensive analysis for adversarial training on the ImageNet dataset. Madry et al. [20] showed the alteration between retraining and projected gradient descent (PGD) attack is one of the most effective methods for adversarial training, in which retraining used the adversarial examples generated by the PGD attack. Tramèr et al. [14] proposed "ensemble adversarial training," in which the training set included adversarial examples produced by the trained models itself and pretrained external models to improve the robustness of the network for the transferred examples.

2.2.2. Network Distillation. As a training strategy, Hinton et al. [21] originally designed a distillation technique that transfers knowledge from a complex network to a simpler network with the purpose of reducing the size of DNNs. For distillation, high temperature can increase the vagueness of softmax output. The property was applied and Papernot et al. [22] further proved that high-temperature softmax decreased the sensitivity of the model to small perturbation. The result is harmful to the adversary as the attack primarily relies on the sensitivity of model. Thus, they proposed defensive distillation to improve the robustness of the model to adversarial examples. In their subsequent work, Papernot and McDaniel [23] solved the numerical instabilities encountered in [22] to extend the defensive distillation method.

2.2.3. Adversarial Examples Reforming. This defense reforms adversarial examples, aiming at mitigating adversarial perturbations prior to the targeted network. Gu and Rigazio [24] proposed a variant of the autoencoder network. At inference, the network is used to encode adversarial examples to remove adversarial perturbations. Santhanam and Grnarova [25] used both the discriminator and the generator of the generative adversarial network (GAN) to project the adversarial examples back onto the legitimate data manifold. In another GAN-based defense method, Samangouei et al. [26] used the generator to sanitize inputs prior to the targeted classifier. Xie et al. [27] randomly resized the adversarial examples and added random padding to the resized examples to reduce the effects of adversarial perturbations. Liao et al. [28] proposed high-level representation-guided denoiser (HGD) to defend the models for image classification. The HGD was trained by optimizing a loss function that represented the difference between the target model's outputs of the clean image and denoised image. Jia et al. [29] studied a preprocessing module to reform adversarial examples, termed as ComDefend, which is composed of a compression convolutional neural network and a reconstruction convolutional neural network.

2.2.4. Adversarial Examples Detecting. The purpose of defense is to filter out adversarial examples. Metzen et al. [30] learned a small network as an auxiliary part of the original network to output the probability of the input example being adversarial. Grosse et al. [31] enabled their model to classify all adversarial examples into one special class by augmenting the targeted network with an additional class. From a Bayesian perspective that the uncertainty of adversarial data is higher than that of legitimate data, Feinman et al. [32] deployed a Bayesian neural network to estimate the uncertainty of input data so as to detect adversarial input data. Xu et al. [33] introduced feature squeezing, an approach to detect adversarial examples by comparing the predictions of the targeted network on the original input and the squeezed input. Fan et al. [12] proposed an integrated detection framework comprising the statistical detector and Gaussian noise injection detector to filter out adversarial examples with different characteristics of perturbations.

2.2.5. Miscellaneous Approaches. Owing to the great diversity of adversarial examples, multiple defense strategies can be integrated to defend against adversarial examples. PixelDefend [34] and MagNet [35] combine an adversarial detector and an adversarial reformer to compose a defense scheme. Akhtar et al. [36] proposed a defense against UAP. They trained a Perturbation Rectifying Network (PRN) as "preinput" layers to a targeted model. The PRN acts as both the UAP detector and input image reformer.

3. Proposed Defense Methodology

3.1. Detection of Adversarial Example Based on Two Detectors. The proposed detection method employs both a statistical detector and minor alteration detector in two consecutive stages to adapt to different perturbation characteristics. Specifically, the tasks of the former and latter detectors are to inspect noticeable and unnoticeable perturbations, respectively. Obviously, the two detectors are complementary. The detection procedure is shown in Figure 3.

3.1.1. Statistical Detector. Unnoticeable perturbations are almost imperceptible to human vision and their statistical characteristics are almost identical to those of legitimate examples. Hence, we construct a positive adversarial example set produced by FGSM, R-FGSM, BIM, and UAP attacks and a negative set including adversarial examples produced by DeepFool, CW_UT, and CW_T attacks, as well as legitimate examples. Inspired by the Subtractive Pixel Adjacency Matrix (SPAM) modeled by Markov process [37], we apply subsets of the transition probability matrix as the perturbation detection feature. Since SPAM is capable of highlighting statistical anomalies, and noticeable perturbations definitely introduce statistical anomalies to an adversarial image, the examples containing noticeable perturbations are naturally expected to be detectable. Last, an ensemble of base fisher linear classifiers undergoes learning to make a decision. During testing, the SPAM-based feature is extracted from the input image and then the ensemble classifier outputs the decision.

(1) SPAM-Based Feature Extraction. $\mathbf{I} = (\mathbf{R}, \mathbf{G}, \mathbf{B})$ represents a color image with a spatial size of $m \times n$, where $\mathbf{R} = \{\mathbf{r}(i,j)\}$, $\mathbf{G} = \{\mathbf{g}(i,j)\}$, and $\mathbf{B} = \{\mathbf{b}(i,j)\}$ denote the red, green, and blue channels, respectively. First, the difference array of the three channels and eight directions are computed. For instance, for the $\mathbf{R}$ channel, the difference array $\mathbf{D}(i,j)_R^{\longrightarrow}$ in the horizontal direction from left-to-right ($\longrightarrow$) is calculated as follows:

$$\mathbf{D}(i,j)_R^{\longrightarrow} = \mathbf{r}(i,j) - \mathbf{r}(i,j+1), \quad 1 \le i \le m,\ 1 \le j \le n-1. \tag{7}$$

The features for the other directions are calculated in the same way. $\beta \in \{\longrightarrow, \leftarrow, \uparrow, \downarrow, \nwarrow, \searrow, \swarrow, \nearrow\}$ denotes eight directions. Then, the second-order Markov process is used to model the second-order SPAM to construct a transition probability matrix. Taking the red channel and the horizontal direction ($\longrightarrow$) as an example, the transition probability matrix is defined using the following formula:

$$\begin{aligned}\mathbf{M}(u,v,w)_R^{\longrightarrow} &= p\big(\mathbf{D}(i,j+2)_R^{\longrightarrow} = u \mid \mathbf{D}(i,j+1)_R^{\longrightarrow} \\ &\qquad = v,\ \mathbf{D}(i,j)_R^{\longrightarrow} = w\big),\end{aligned} \tag{8}$$

where $-T \le u, v, w \le T$, T is a preset parameter and p denotes a conditional probability which satisfies $\mathbf{D}(i,j+2)_R^{\rightarrow} = u$ conditioned on $\mathbf{D}(i,j+1)_R^{\longrightarrow} = v$ and $\mathbf{D}(i,j)_R^{\rightarrow} = w$. Since u, v, and w change from $-T$ to T, the range of each of the parameters is $2T+1$ and the size of transition probability matrix $\mathbf{M}$ is $(2T+1)^3$, accordingly. The transition probability matrices of the other directions and other channels can be defined in the same way. To reduce the dimension, the transition probability matrices of some directions are fused on average as follows:

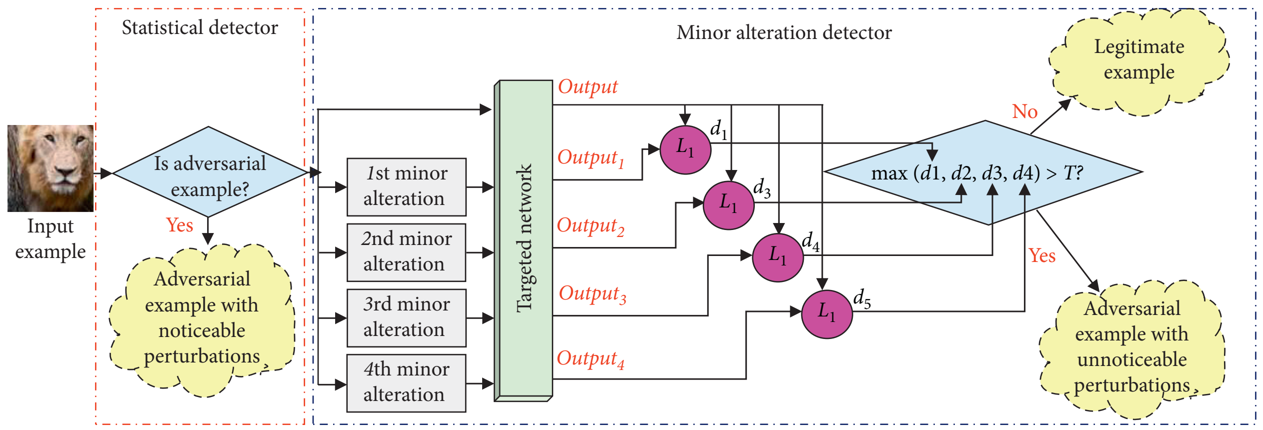

Figure 3: An overview of detection of adversarial example. First, the input example is fed into the statistical detector. If the input example is not determined to be an adversarial example with noticeable perturbations, it will be further analyzed by the minor alteration detector. Specifically, the input example is altered by four minor operations and then the original input and its four altered counterparts are all fed into the targeted network. Then, the L_1 norm difference between two outputs corresponding to the original input and any one of the four alterations is calculated. Finally, the max value of the four differences is compared with a threshold T. If the maximum exceeds the threshold, the input example will be detected as adversarial example with unnoticeable perturbations, otherwise legitimate example.

$$\begin{aligned}\mathbf{F}(u,v,w)_R^1 &= \frac{1}{4}\left[\mathbf{M}(u,v,w)_R^{\rightarrow} + \mathbf{M}(u,v,w)_R^{\leftarrow} + \mathbf{M}(u,v,w)_R^{\uparrow} + \mathbf{M}(u,v,w)_R^{\downarrow}\right],\\ \mathbf{F}(u,v,w)_R^2 &= \frac{1}{4}\left[\mathbf{M}(u,v,w)_R^{\nwarrow} + \mathbf{M}(u,v,w)_R^{\searrow} + \mathrm{M}(u,v,w)_R^{\swarrow} + \mathbf{M}(u,v,w)_R^{\nearrow}\right].\end{aligned} \tag{9}$$

The fused transition probability matrix is used as the feature. Finally, the features of the three channels are denoted as $\mathbf{F}_R = \left[\mathbf{F}_R^1\ \mathbf{F}_R^2\right]$, $\mathbf{F}_G = \left[\mathbf{F}_G^1\ \mathbf{F}_G^2\right]$, and $\mathbf{F}_B = \left[\mathbf{F}_B^1\ \mathbf{F}_B^2\right]$ (the parameters "u, v, w" are omitted). The final feature $\mathbf{F}_M$ is a joint of F_R, $\mathbf{F}_G$, and $\mathbf{F}_B$. The dimension of the features of each channel is $2(2T+1)^3$. To obtain an optimal trade-off between efficiency and performance, we set T as 3 and the dimension of $\mathbf{F}_M$ is $686 * 3 = 2058$.

To provide an intuitive understanding of SPAM-based feature extraction, a legitimate image and its corresponding FGSM attacked adversarial image were selected to illustrate the significant differences between their $\mathbf{F}_M$ features (see Figure 4).

(2) Ensemble Binary Classifiers. Ensemble classifiers [38] consist of multiple base classifiers independently trained from a set of positive and negative samples. As a base classifier, each fisher classifier is trained from a random subspace of the entire feature space. The symbol L stands for the number of base classifiers. For the ith base classifier, the corresponding random subspace are represented using $\mathscr{D}_i, i = 1, \ldots, L$. Then, we train a base classifier $\mathscr{B}_i$ on features of positive and negative samples using the fisher linear discriminant (FLD). For a test feature $\mathbf{y}$, the decision of the ith base learner is $\mathscr{B}_i(\mathbf{y}^{(\mathscr{D}_i)})$. After collecting all L decisions, the final classifier output $B(\mathbf{y})$ is formed by combining them using a majority voting strategy, where 1 stands for positive class and 0 for negative class:

$$B(\mathbf{y}) = \begin{cases} 1, & \text{when } \sum_{i=1}^{L} \mathscr{B}_i\left(\mathbf{y}^{(\mathscr{D}_i)}\right) < \dfrac{L}{2}, \\ 0, & \text{when } \sum_{i=1}^{L} \mathscr{B}_i\left(\mathbf{y}^{(\mathscr{D}_i)}\right) > \dfrac{L}{2}, \\ \text{random}, & \text{otherwise.} \end{cases} \tag{10}$$

3.1.2. Minor Alteration Detector.

Although the adversarial examples with noticeable perturbations are filtered out with a nearly perfect effect by the statistical detector, most of the adversarial examples with unnoticeable perturbations remain undetected. Thus, this section is concerned with detecting adversarial examples corrupted with unnoticeable perturbations.

We notice that if a legitimate example undergoes a minor alteration, the classification result given by the targeted network would be relatively unchanged, whereas an adversarial example containing unnoticeable perturbations after minor transformation would have a significant effect on the classification result. The reason may be that the legitimate example is located on the manifold of its GT class such that a slight bias would not influence the result critically. Nevertheless, an adversarial example with unnoticeable perturbations would still be close to the manifold of its GT class, but these perturbations are easily damaged by minor transformations. Thus, its classification result would most

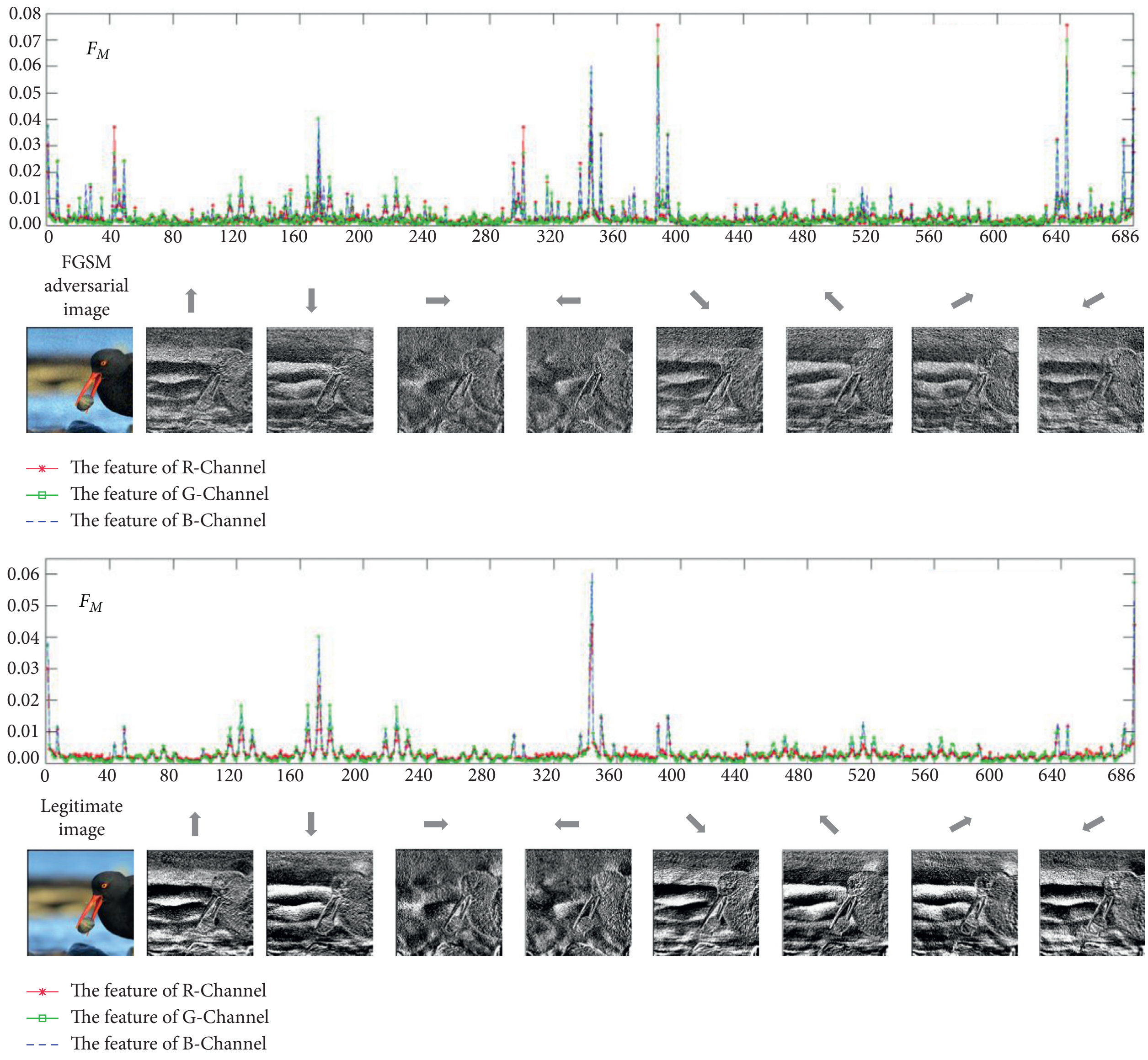

FIGURE 4: The SPAM feature $\mathbf{D}(i, j)_R$ in eight directions and the final feature $\mathbf{F}_M$ are illustrated. The eight directions are expressed as $\{\uparrow, \downarrow, \longrightarrow, \leftarrow, \searrow, \nwarrow, \nearrow, \swarrow\}$. The difference between the FGSM adversarial example and legitimate example are distinctly observed.

probably change once it undergoes minor alterations. Depending on this observation and the need to adapt to a range of adversarial perturbation characteristics, we devise four minor alterations to images and a straightforward yet effective *max* fusion rule: taking the max value among four *output differences* between the image before and after alteration and the output is given by the targeted network. In terms of the max value, the detector can make a decision. If the max value is greater than a threshold, the input sample is classified as an adversarial example with unnoticeable perturbations; otherwise, it is a legitimate example. This detection process is referred to as a minor alteration detector (see “Minor alteration detector” in Figure 3). Obviously, the statistical detector and the minor alteration detector function in a complementary way to detect adversarial examples.

Specifically, the four alterations operate as follows:

remove&interpolation (*ri*). Remove a small number of rows and columns at fixed positions in the image and then recover the removed rows and columns by interpolation (see Figure 5(a)).

remove&expansion (*re*). Remove a small peripheral part and then expand the remaining part to the original image size (see Figure 5(b)).

rotate-clockwise (*rc*). Rotate clockwise at a small angle around the geometric center of the image.

rotate-anticlockwise (*ra*). Rotate anticlockwise at a small angle around the geometric center of the image.

Computing the difference is another critical issue, we choose a prediction probability distribution vector as the output and use the ℓ_1 norm to measure the output difference $d(\mathbf{x}, \mathbf{x}_{\text{alt}})$:

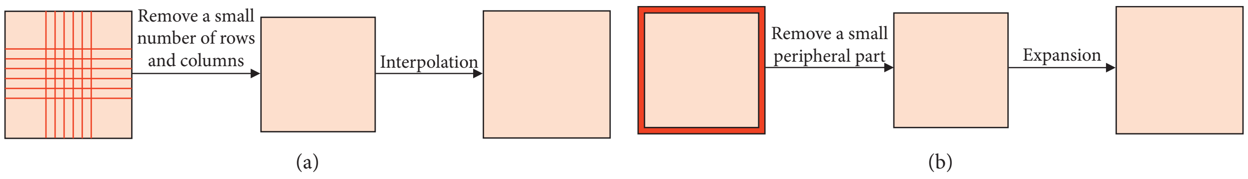
Remove a small number of rows and columns
Interpolation
(a)
Remove a small peripheral part
Expansion
(b)

FIGURE 5: Illustration of *ri* and *re* alterations.

$$d(\mathbf{x}, \mathbf{x}_{\text{alt}}) = \left\| g(\mathbf{x}) - g(\mathbf{x}_{\text{alt}}) \right\|_1, \tag{12}$$

where $g(\mathbf{x})$ and $g(\mathbf{x}_{\text{alt}})$ denote the prediction probability distribution vectors of targeted network for the original input $\boldsymbol{x}$ and its minor alteration version $\mathbf{x}_{\text{alt}}$, respectively. Furthermore, the range of $d(\mathbf{x}, \mathbf{x}_{\text{alt}})$ is from 0 to 2, with a higher value indicating a greater difference. We expect the difference to be as small as possible for legitimate input and as large as possible for the adversarial input. Figure 6 shows some examples of a legitimate image and adversarial images produced by DeepFool, CW_UT, and CW_T attacks. Four altered versions of each image are demonstrated. Both the corresponding output difference and the max value are coincident with our expectations. Figure 7 presents seven histograms of $d(\mathbf{x}, \mathbf{x}_{\text{alt}})$ for both legitimate examples (red) and adversarial examples (blue) crafted by the aforementioned attacks. In the case of the DeepFool, CW_UT, and CW_T attacks, the difference distributions of legitimate and adversarial examples can be separated well, whereas for FGSM, R-FGSM, BIM, and UAP attacks, the difference distributions considerably overlap. These results indicate that noticeable perturbations are relatively hard to be damaged, which is probably attributed to the large distances of these adversarial examples from their GT manifolds.

3.2. The Deep Residual Generative Network to Clean Adversarial Perturbations. Cleaning adversarial perturbations is also a feasible defense scheme. In this paper, we utilize our previously proposed network called ResGN to reform adversarial examples. The network with residual blocks is conditionally generative and is trained in a supervised way. The supervisions are pairs of legitimate image and corresponding adversarial image, and the adversarial images are generated by white box attacking on a certain targeted network. The optimization of ResGN is driven by minimizing a joint loss composed of pixel loss, texture loss, and semantic loss, in which the latter two losses depend on a pretrained network independent of the targeted network (see Figure 8). The specific structure of ResGN and its detailed training algorithm are referred in [10].

3.3. The Complete Defense Framework. Thus far, we have discussed two defense methods: adversarial examples detection and adversarial perturbations cleaning. Accordingly, we can implement three defense patterns: the integration of a nonadversarially trained targeted network with adversarial examples detection, adversarial perturbations cleaning, and both of them (see Figures 9(a)~9(c)). Apart from the three patterns, an adversarially trained targeted network is a well-known defensive technique [10], in which the input samples are directly classified by such a targeted network (see Figure 9(d)). Naturally, other defensive options are to replace the nonadversarially trained targeted network in Figures 9(a)~9(c) with its adversarially trained counterpart to produce three other defense patterns (see Figures 9(e)~9(g)). Usually, we refrain from choosing combination 9(a) and 9(e) as the rejection of input samples is not allowed in some scenarios. More importantly, the detection module can prevent most of the legitimate examples from being cleaned because samples detected to be legitimate would bypass the cleaning module. Detection is especially meaningful if legitimate samples form a large percentage of input samples. However, adversarial examples would also elude detection. Fortunately, the adversarially trained network can alleviate the problem because of its strong robustness to adversarial examples. Moreover, cleaned legitimate examples slightly deviate from their original versions and it is not always possible to completely eliminate adversarial perturbations because of the diversity of perturbations. These issues suggest the use of an adversarially trained targeted network rather than one that is nonadversarially trained. In addition, the joint use of detection and cleaning modules is expected to significantly boost the performance of the adversarially trained targeted network alone. This led us to combine the two proposed detectors, the ResGN, and an adversarially trained targeted network and to construct a complete defense framework (see Figure 9(g)).

4. Results and Discussion

4.1. Experiment Setup. In our experiments, we used a PC equipped with an i7-6850K 3.60 GHz CPU and a NVIDIA TITAN X GPU. The developing environment is TensorFlow [39]. We chose Inception-v3 [40] and adv-Inception-v3 [11] as targeted network and adversarially trained network, respectively. From the ImageNet validation dataset, 5000 legitimate images correctly classified by the pretrained Inception-v3 model were selected, in which 4000 images form the training set and 1000 images constitute the testing set. All adversarial examples were generated from the legitimate images using attacking implementation from Cleverhans library [41]. Table 1 demonstrates the parameters of these attacking methods that are mentioned in Section 2.1, and Table 2 shows the classification accuracy of their corresponding adversarial examples using Inception-v3.

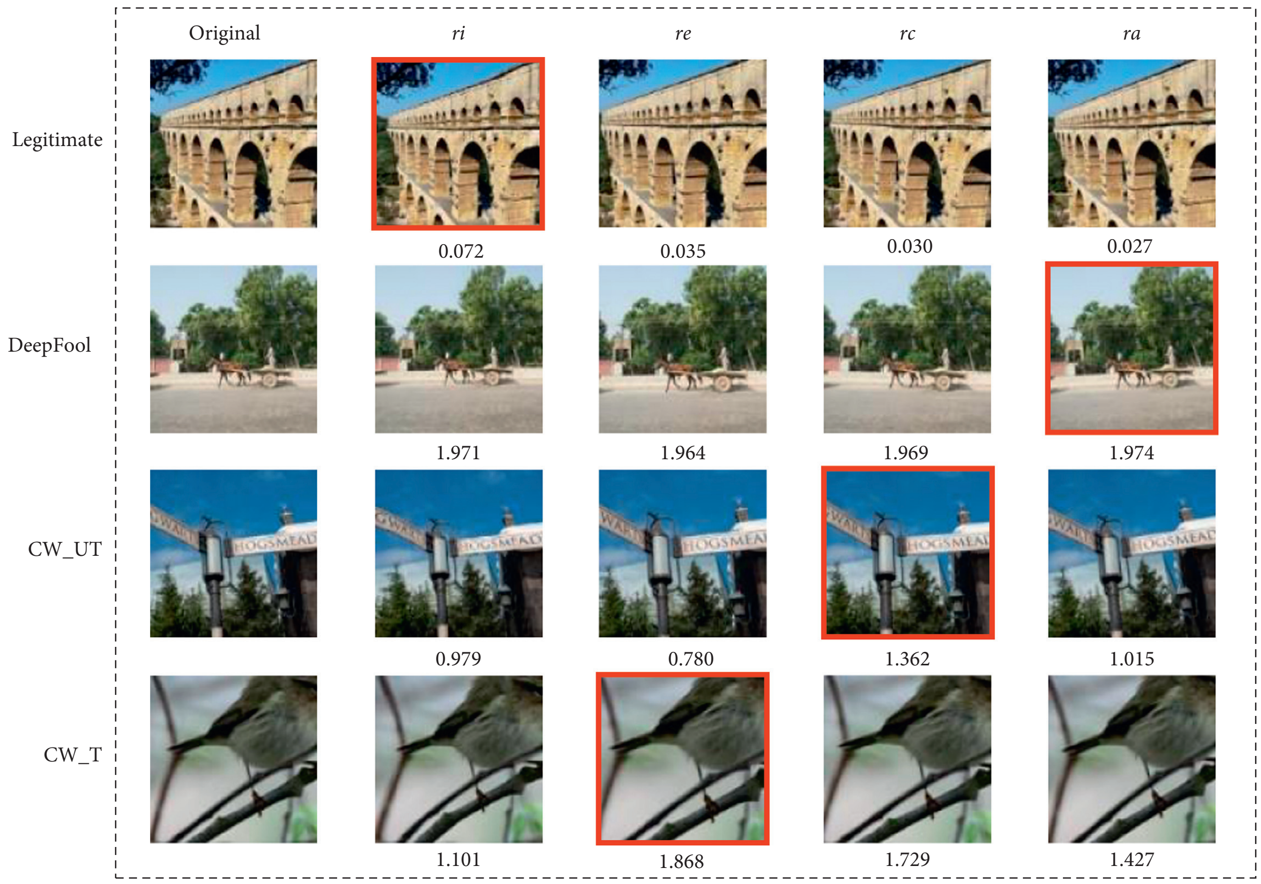

Figure 6: Minor alterations and output difference demonstration. The first column shows four original images including a legitimate image and three adversarial images crafted by DeepFool, CW_UT, and CW_T. Each of the remaining four columns are corresponding minor alteration images for the four original images. The value below each image is the difference between the outputs of it and its original one using the targeted network (Inception-v3). The image marked with the red box has obtained the max value among the four differences in each row.

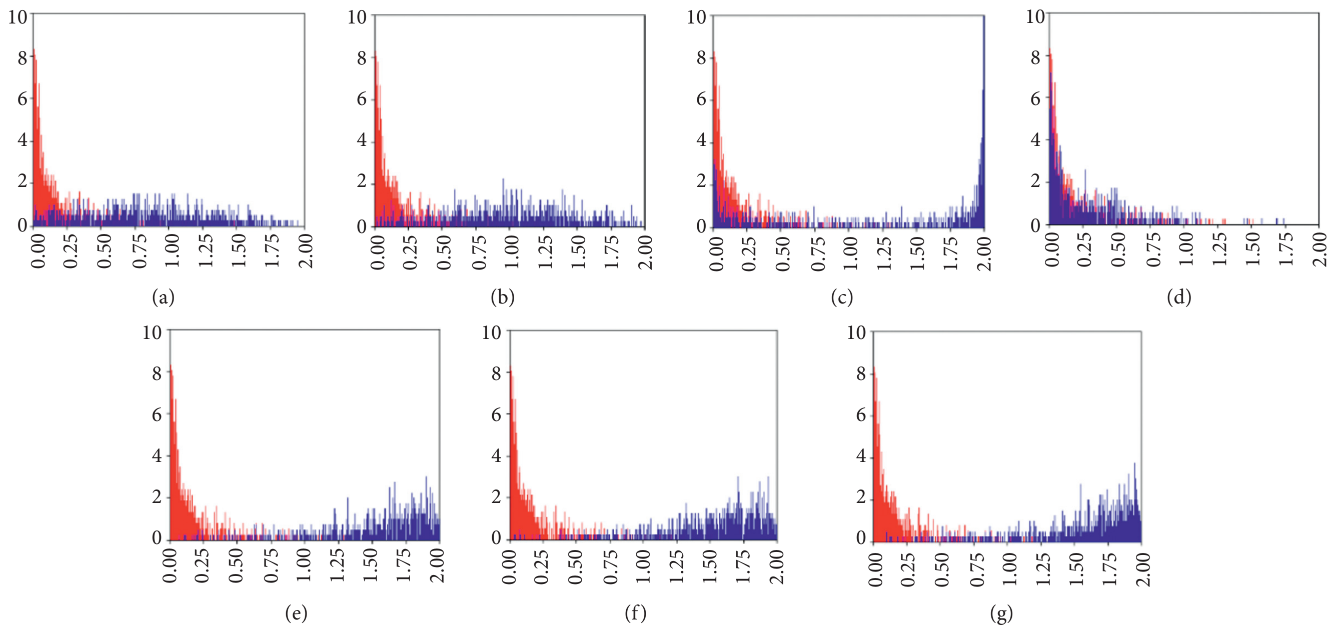

Figure 7: Minor alteration detector difference histograms using max fusion rule for legitimate examples (red) and adversarial examples (blue) generated by FGSM, R-FGSM, BIM, UAP, DeepFool, CW_UT, and CW_T on the training set. The horizontal axis represents the distance between the two vectors of the original input and its corresponding alteration version output by the targeted network (Inception-v3). The vertical axis represents the number of images at a certain distance. (a) FGSM examples. (b) R-FGSM examples. (c) BIM examples. (d) UAP examples. (e) DeepFool examples. (f) CW_UT examples. (g) CW_T examples.

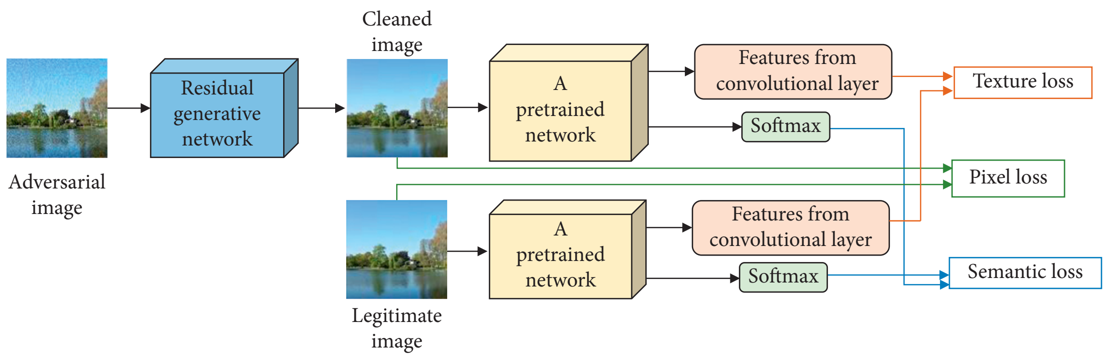

Figure 8: The training architecture of ResGN. The optimization of ResGN is driven by minimizing a joint loss containing pixel loss, texture loss, and semantic loss. The latter two losses are provided by any pretrained network.

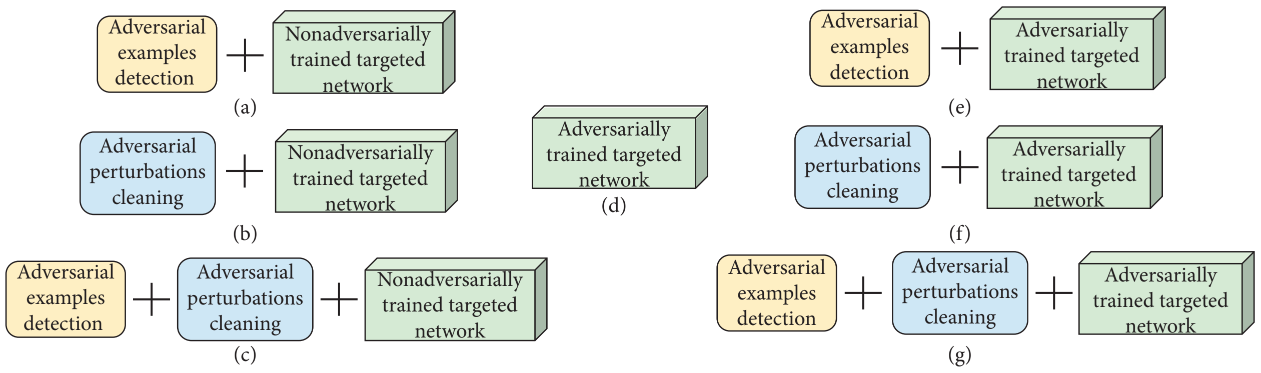
Adversarial examples detection + Nonadversarially trained targeted network
(a)
Adversarial perturbations cleaning + Nonadversarially trained targeted network
(b)
Adversarial examples detection + Adversarial perturbations cleaning + Nonadversarially trained targeted network
(c)
Adversarially trained targeted network
(d)
Adversarial examples detection + Adversarially trained targeted network
(e)
Adversarial perturbations cleaning + Adversarially trained targeted network
(f)
Adversarial examples detection + Adversarial perturbations cleaning + Adversarially trained targeted network
(g)

Figure 9: Seven defense patterns.

Table 1: The parameters of evaluated attacks.

Attacks	FGSM	R-FGSM	BIM	UAP	DeepFool	CW_UT	CW_T
ϵ	32/255	32/255	32/255	0.1	—	—	—
α	——	0.01	1	—	—	—	—
δ	—	—	—	0.8	—	—	—
Iterations	—	—	20	400	200	300	300

Table 2: The classification accuracy of adversarial examples generated by evaluated attacks on the testing set (%).

Attacks	FGSM	R-FGSM	BIM	UAP	DeepFool	CW_UT	CW_T
Accuracy	26.1	12.8	0	3.5	0.2	0.7	1.0

4.2. Results of Adversarial Example Detection

4.2.1. Results of Statistical Detector. This experiment was designed to assess the ability of the statistical detector. First, we construct seven positive training sets containing the adversarial examples produced, by using an individual attacking method and the legitimate examples to compose a common negative training set. For FGSM, R-FGSM, and BIM attacks, the attacking strength is $\epsilon = 32/255$ for both the training and testing sets. Table 3 lists the performance of the seven statistical detectors on the testing set in terms of True Positive Rate (TPR) and False Positive Rate (FPR). The results indicate that TPR of the adversarial examples generated by FGSM, R-FGSM, BIM, and UAP is close to 100% and FPR of legitimate examples is below 1%. We can conclude that the adversarial examples crafted by the four attacks were clearly distinguished from the legitimate examples. However, for the detectors that the positive set is composed of adversarial examples crafted by DeepFool, CW_UT, or CW_T, both TPR and FPR are not satisfied. The results imply that SPAM-based feature is not appropriate to characterize the unnoticeable perturbations.

Table 3: Performance of seven statistical detectors (%).

Positive examples	FGSM	R-FGSM	BIM	UAP	DeepFool	CW_UT	CW_T
Negative examples				Legitimate			
TPR	99.8	100	99.8	99.4	85.8	81.9	81.1
FPR	0.5	0.2	0.9	0.6	26.0	25.5	26.6

These results motivate us to attempt to separate all samples into two groups in terms of perturbation significance: one group includes FGSM, R-FGSM, BIM, and UAP adversarial examples with noticeable perturbations and the other group contains the adversarial examples with unnoticeable perturbations produced by DeepFool, CW_UT, CW_T, and legitimate examples. Accordingly, the former and the latter groups constitute the positive and negative training set, respectively. Table 4 lists overall TPR and FPR values of the testing set which are 99.6% and 0.6%, respectively. These results confirm that the statistical detector achieves promising performance.

Parameter ϵ in FGSM, R-FGSM, and BIM attacks controls the attacking strength. In a real setting, attackers may yield adversarial examples using various attacking strengths. Thus, we designed this experiment to explore the transfer ability of the statistical detector. The results in Table 5 show that each detector learned from examples with ϵ performs well on the testing set when ϵ is the same or larger, whereas the performance decreases when ϵ is weaker for the testing set. Thus, from the average sight, the detector learned from samples set with $\epsilon = (8/255)$ performs the best (99.4%) and is supposed to have the best transfer ability. The results validate that more strongly attacked examples are much easier to detect since the statistical anomaly hidden in them is more evident than that in more weakly attacked examples. Although higher transfer ability is obtained at the cost of a slightly increased FPR (1.7%), we still favor the detector learned from the sample set with $\epsilon = 8/255$ as the ultimate statistical detector owing to its satisfactory TPR.

4.2.2. Results of Minor Alteration Detector. The performance of the statistical detector in terms of detecting the adversarial examples produced by FGSM, R-FGSM, BIM, and UAP is significantly high. Unfortunately, it was not possible to reliably distinguish the adversarial examples produced by DeepFool, CW_UT, and CW_T from legitimate examples. This experiment was therefore intended to evaluate the capability of the minor alteration detector to detect the three types of adversarial examples with unnoticeable perturbations.

First, we selected an optimal parameter for each alteration from the candidate optional multiple parameters. We calculated the AUC value of the ROC curve (ROC-AUC) for all candidate parameters (see Table 6). The value in bold indicates the top value and the corresponding parameter. Then, an optimal decision threshold is needed to be determined. Because a detector with high TPR at the cost of high FPR is meaningless, we chose 5% as the acceptable FPR. Thus, the optimal decision threshold is the value corresponding to 5% FPR. Note that the optimal parameters for all alterations and the final optimal decision value are derived from training set. The performance of the different single alterations is also compared by evaluating TPR and FPR for the four alterations, in addition to the max fusion rule. Table 7 lists that all corresponding thresholds and the optimal TPR (94.8%) and FPR (4.7%) have been obtained by the max fusion rule. These results confirm that the max fusion rule has an advantage over the single alterations.

4.2.3. Results of Combination of Two Detectors. The combined results show that the combination of the statistical detector and the minor alteration detector (shown in Figure 3) enables all seven types of adversarial examples to be detected. A promising trade-off between TPR (97.6%) and FPR (6.3%) (Table 8) was obtained by combining the two detectors.

We next compare the performance of the combination of the two detectors with that of the integrated detector [12] and feature squeezing detector [33]. The proposed detector achieves highest TPR on adversarial examples produced by FGSM, R-FGSM, BIM, and UAP. However, the performance of the proposed detector on adversarial examples produced by DeepFool, CW_UT, and CW_T is slightly weaker than that of the integrated detector. Although feature squeezing detector achieves a higher TPR on CW_UT and CW_T adversarial examples than ours, the proposed detector has better performance on the other types of adversarial examples. It is worth mentioning that the proposed detector obtained a lower FPR (6.3%). In general, the performance of the proposed detector in terms of detecting adversarial example is satisfactory.

4.3. Results of Optimization of ResGN. We discovered that increasing the number of residual blocks improves the performance of ResGN; nevertheless, it is at the expense of a considerable increase in the computational complexity. Considering the need to maintain a balance, we use 24 residual blocks in ResGN. Considering the adaptability of ResGN, vgg-19 is selected as the pretrained network during training rather than Inception-v3 or adv-Inception-v3 network. Last, adversarial examples produced by FGSM attacking method with strength $\epsilon = 32/255$ compose training set and corresponding legitimate images are GT. A set of images including four legitimate images, 16 pairs of adversarial images, and corresponding cleaned images by optimized ResGN are shown in Figure 10. It can be seen that, in addition to the adversarial examples crafted by FGSM and an attacking strength of 32/255, adversarial examples other than those obtained with FGSM or an attacking strength of 32/255 also reveal satisfactorily cleaned visual effects. The results in Table 9 confirm that ResGN optimized by FGSM

Table 4: Performance of the single statistical detector (%).

Positive examples	FGSM	R-FGSM	BIM	UAP	Overall
TPR	99.9	100	100	98.3	**99.6**
Negative examples	DeepFool	CW_UT	CW_T	Legitimate	Overall
FPR	0.6	0.6	0.6	0.7	**0.6**

Table 5: Evaluation for transfer ability of the statistical detector for various attacking strengths.

Train	Test					
	TPR (%)					FPR (%)
	$\epsilon = 8/255$	$\epsilon = 16/255$	$\epsilon = 24/255$	$\epsilon = 32/255$	Average	
$\epsilon = 8/255$	98.6	99.5	99.6	99.7	**99.4**	1.7
$\epsilon = 16/255$	93.6	99.0	99.4	99.5	97.9	1.5
$\epsilon = 24/255$	86.1	97.1	99.3	99.5	95.5	1.4
$\epsilon = 32/255$	80.4	95.1	98.2	99.6	93.3	0.6

Table 6: AUC of ROC curve for minor alteration using various parameters on the training set.

Operation			*ri*		
Parameter	30	40	50	**60**	70
ROC-AUC	0.9804	0.9801	0.9827	**0.9839**	0.9818
Operation			*re*		
Parameter	5	10	**15**	20	25
ROC-AUC	0.9738	0.9775	**0.9781**	0.9779	0.9753
Operation			*rc*		
Parameter	1	**2**	3	4	5
ROC-AUC	0.9719	**0.9793**	0.9782	0.9771	0.9766
Operation			*ra*		
Parameter	1	**2**	3	4	5
ROC-AUC	0.9717	**0.9796**	0.9793	0.9789	0.9777

Table 7: Performance of the minor alteration detector (%).

	Operation	*ri*	*re*	*rc*	*ra*	*max*
	Threshold	0.747	0.682	0.574	0.564	0.993
TPR	DeepFool	91.8	93.4	90.8	91.5	**94.7**
	CW_T	92.3	94.1	91.2	92.4	**94.6**
	CW_UT	92.7	94.1	91.3	92.7	**95.0**
	Overall	92.3	93.9	91.1	92.2	**94.8**
FPR	Legitimate	5.7	6.1	5.5	6.0	**4.7**

Table 8: Performance of the combination of the two detectors vs. the integrated detector [12] and feature squeezing detector [33] (%).

Detector	TPR								FPR
	FGSM	R-FGSM	BIM	UAP	DeepFool	CW_UT	CW_T	Average	
The proposed detector	99.8	99.9	99.9	99.5	94.7	94.6	95.1	97.6	**6.3**
Integrated detector [12]	99.5	99.6	99.5	96.9	96.3	96.1	96.4	97.7	7.1
Feature squeezing detector [33]	56.3	52.3	67.9	88.1	90.4	95.3	97.1	78.2	8.2

adversarial examples with an attacking strength of 32/255 has better performance on adversarial examples.

4.4. Results of the Complete Defense Framework and Comparison with Other Defense Methods. As stated in Section 3.3, adversarial example detection, adversarial perturbation cleaning, and an adversarially trained network form a complete defense framework which is the most powerful defense pattern. This experiment aims to validate this point. In addition, to observe the relation between the performance

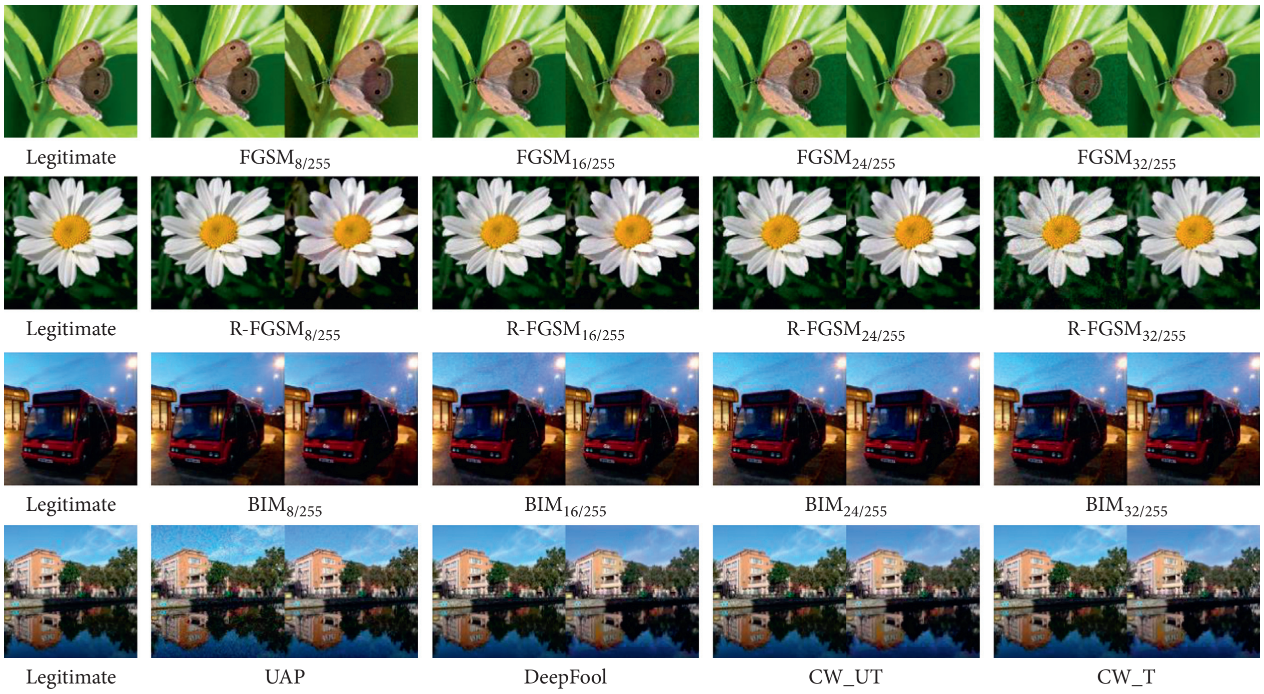

Figure 10: A set of images including four legitimate images, 16 pairs of adversarial images (a), and corresponding cleaned images by ResGN (b).

Table 9: Performance of ResGN optimized by FGSM adversarial examples with strength $\epsilon = 32/255$.

Training set	Testing set					
			FGSM			
		$\epsilon = 8/255$		$\epsilon = 16/255$	$\epsilon = 24/255$	$\epsilon = 32/255$
		80.6		86.3	89.9	93.6
			R-FGSM			
		$\epsilon = 8/255$		$\epsilon = 16/255$	$\epsilon = 24/255$	$\epsilon = 32/255$
FGSM $\epsilon = 32/255$		78.8		81.4	87.2	92.3
			BIM			
		$\epsilon = 8/255$		$\epsilon = 16/255$	$\epsilon = 24/255$	$\epsilon = 32/255$
		79.3		90.6	86.8	91.2
	UAP	DeepFool	CW_UT	CW_T	Legitimate	Average
	90.4	90.6	91.7	90.4	98.1	87.6

and the proportion of legitimate examples in the testing set, we constructed multiple testing sets composed of various proportions of legitimate examples and fixed the total number of testing samples at 1000. The proportion was increased in increments of 10 percent. The adversarial examples contained in each of the testing sets vary with respect to their type and attacking strength. Furthermore, we compared the performance of our complete method with that of the RADOMIZATION [27], HGD [28], and ComDefend [29] techniques. In Figure 11, the "accuracy" is plotted as a function of the "proportion" for the three forms of defense, details of which are provided in the legend. Instead of Inception-v3, adv-Inception-v3 was used as targeted network and its ability to function alone as a form of defense is shown in Table 10. The experimental results indicate that, for detection alone or for a combination of detection and ResGN, working collaboratively with Inception-v3 or adv-Inception-v3 yields performance superior to that of other defense methods; second, the combination of our proposed defense methods with adv-Inception-v3 improved the performance of adv-Inception-v3 alone remarkably (see Table 10). These results verify that ResGN is actually able to meaningfully improve the performance of an adversarially trained network because it is at least capable of mitigating adversarial perturbations even though the perturbations are impossible to remove perfectly owing to their diversity. Finally, the performance of the joint use of detection and ResGN with the targeted network is expected to outperform that of ResGN alone with the targeted network when the proportion of legitimate examples is relatively large. Fortunately, most of the input samples are legitimate in a real setting; hence, the complete defense framework is actually critical for counteracting adversarial examples.

We discuss the computational complexity of the proposed complete defense framework. Suppose that there are N samples in total for testing and the percent of adversarial examples is s. We further assume the number of detected adversarial examples by the detector is very close to the

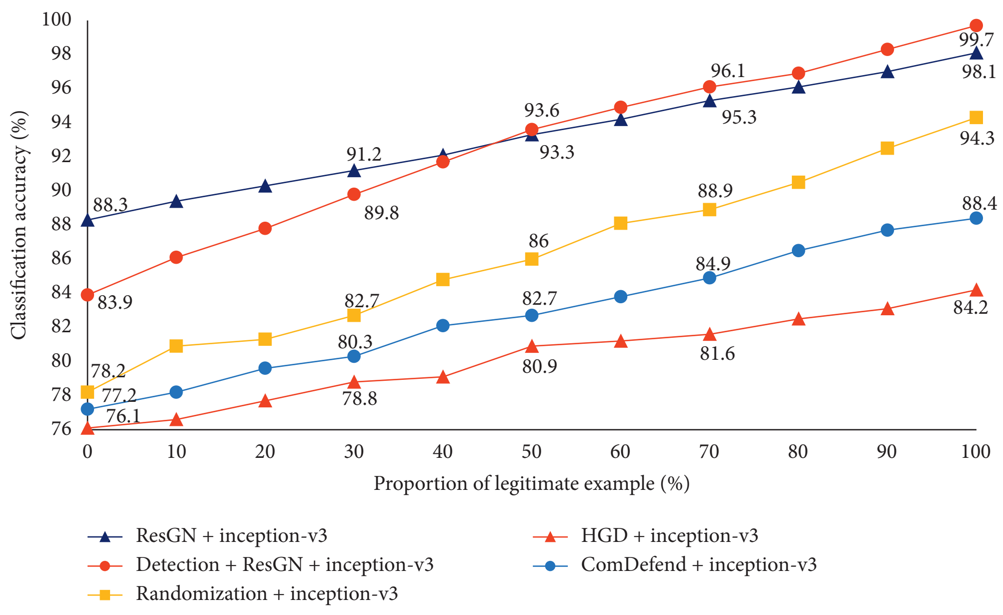

Figure 11: Classification accuracy of Inception-v3 as targeted network.

Table 10: Classification accuracy of adv-Inception-v3 as targeted network (%).

The proportion of legitimate examples (%)	0	10	20	30	40	50	60	70	80	90	100	Average
Adv-inception-v3	89.9	90.6	91.2	92.0	92.9	93.6	94.4	95.1	96.4	96.9	97.3	93.7
ResGN + Adv-inception-v3	98.4	98.5	98.7	98.8	99.0	99.1	99.3	99.3	99.4	99.5	99.8	99.1
Detection + ResGN + Adv-inception-v3	98.2	98.4	98.7	98.9	99.1	99.2	99.4	99.5	99.7	99.8	99.9	**99.2**
Randomization + Adv-inception-v3	89.3	89.7	90.3	91.7	92.3	93.1	93.8	94.3	95.1	95.8	96.2	92.9
HGD + Adv-inception-v3	84.3	84.9	85.8	86.4	87.5	88.3	89.2	90.1	90.5	91.1	92.1	88.2
ComDefend + Adv-inception-v3	89.1	89.6	90.3	90.5	91.1	91.3	91.9	92.3	92.5	93.4	94.6	91.5

number of genuine adversarial examples. T_d, T_c, and T_r denote the average time of detection, the average time of adversarial perturbations cleaning, and the average time of recognition for each sample, respectively. Then, for the proposed complete defense, the total time required for recognizing N samples is $N \times (T_d + s \times T_c + T_r)$. For the defense excluding detection, the total time required for recognizing N samples is $N \times (T_c + T_r)$. For the complete defense, we assume a special case. If the s equals to 0, the required time is $N \times (T_d + T_r)$. So, in this case, if the detection is more efficient than adversarial perturbation cleaning, which means $T_d < T_c$, the complete defense will be more efficient than the defense excluding detection, and vice versa. In addition, with s increasing, the required time will increase. In sum, the computational complexity of the complete defense depends on two factors: the computational complexity of the detection module and the percent of the adversarial example. The percent of adversarial example is out of control, so the efficiency of detection is crucial. For each image, the average required time of our proposed framework is 1.5 seconds. Specially, the detection module spends 1.0 seconds. Therefore, the circumstance that the detection module in our work has a heavier time consumption than adversarial perturbation cleaning module, the adversarially trained targeted network calls for a study of more efficient detectors in the future.

4.5. The Complete Defense Aware Attack. We evaluate the robustness of the complete defense framework in totally white box setting. The adversary is aware of the complete defense framework and includes them in generating adversarial examples. The entire attacking process is explained in Algorithm 1. We consider two types of white box attacks, which are single-step attack FGSM and iterative attack BIM. The adversarial examples generated by using them to attack the Inception-v3 and complete defense framework are illustrated in Figure 12. More serious color or texture distortions are induced by attacking the complete defense than sole Inception-v3, and the differences could be observed for FGSM and BIM from the global and local region level. Table 11 shows the success rates of FGSM and BIM attacking Inception-v3 and complete defense framework on the testing set. The success rates of attacking complete defense framework are much lower than that of Inception-v3. In terms of our proposed complete defense itself, the success rate of a single-step attack is higher than that of an iterative attack. The adversary almost exclusively attacks the adversarially trained targeted network using single-step attack due to the involvement of detection. While in iterative attack, it is essential for adversary to attack the combination of adversarial perturbation cleaning and an adversarially trained targeted network. The results confirm that the

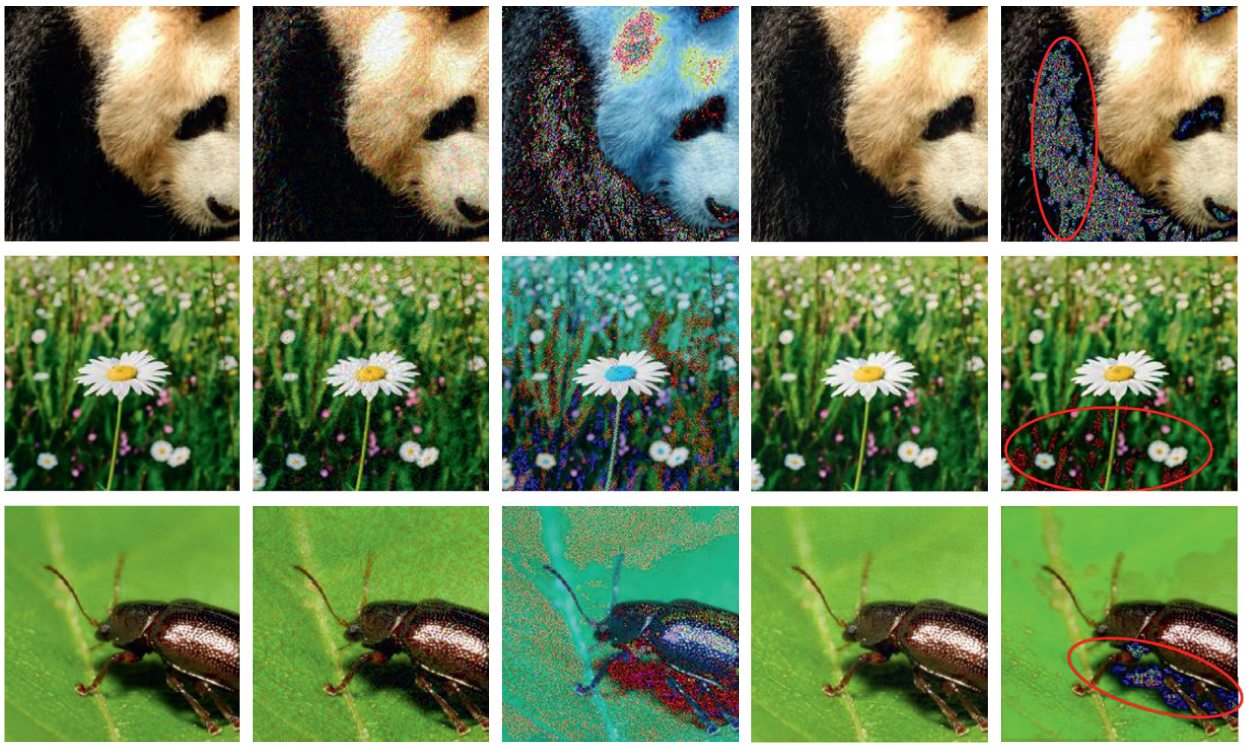

Figure 12: The legitimate examples are shown in the left column. (a) The adversarial examples are generated by using FGSM to attack Inception-v3. (b) The adversarial examples are generated by using FGSM to attack proposed complete defense framework. (c) The adversarial examples are generated by using BIM to attack Inception-v3. (d) The adversarial examples are generated by using BIM to attack proposed complete defense framework. More serious color or texture distortions are induced by attacking the complete defense than sole Inception-v3, and the differences could be observed for FGSM from global level (see (b)) and BIM from local region level (see (d)). The differences in local region are marked with the red circle.

```
Input: a legitimate image $\mathbf{x}_L$; adversarial example detection, adversarial perturbation cleaning and an adversarially trained targeted network denote as $\Phi_d$, $\Phi_{|c|}$, and $\Phi_{\text{adv}}$, respectively.
Parameter: N: the max attack iterations (N = 1 especially for single-step attack)
  Initialize $\hat{x} = x_L$
  for $k = 1, \ldots, N$ do
  (i) $\Phi_d$ decides $\hat{x}$ whether legitimate or adversarial image
   (i) If the decision is legitimate image, $\hat{x}$ is updated by attacking $\Phi_{\text{adv}}$
  (ii) If the decision is adversarial image, $\hat{x}$ is updated by attacking the combination of $\Phi_c$ and $\Phi_{\text{adv}}$
  end
Output an adversarial image $\mathbf{x}_{\text{adv}} = \hat{\mathbf{x}}$
```

Algorithm 1: The complete defense aware attack.

Table 11: The success rates of FGSM and BIM attacking the Inception-v3 and complete defense framework (%).

Attack	FGSM		BIM	
Targeted network	Inception-v3	Complete defense	Inception-v3	Complete defense
Success rate	73.9	**28.9**	100	**13.7**

complete defense maintains it robustness in totally white box setting.

5. Conclusions

We propose a complete defense framework comprising three modules: adversarial example detection, adversarial perturbation cleaning, and adversarially trained targeted network. Specifically, if an input sample is detected to be adversarial, the sample is cleaned by ResGN and then classified by the adversarially targeted network. Otherwise, the sample is directly classified by the adversarially targeted network. Furthermore, detection is accomplished by two complementary detectors adaptive to adversarial perturbation characteristics: the statistical detector filters out the adversarial examples with noticeable perturbations and the minor alteration detector filters out the adversarial examples with unnoticeable perturbations. In future work, the proposed complete defense framework is expected to extend to other applications, such as face recognition. Furthermore, we aim to dynamically optimize the proposed defense method to incrementally boost the capability to counteract adversarial examples.

Data Availability

The ImageNet dataset used in the experiments is public. Please refer to the corresponding project website for downloading these datasets. The source code of the proposed method is available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

We would like to thank Editage (http://www.editage.cn) for English language editing. This work was supported by the Natural Science Foundation of Shanghai under Grant no. 16ZR1411100 and Shanghai Committee of Science and Technology under Grant no. 19511105503.

References

[1] Z. Zhao, P. Zheng, S. Xu, and X. Wu, "Object detection with deep learning: a review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[2] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing (review article)," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[3] L. Deng, J. Li, J.-T. Huang et al., "Recent advances in deep learning for speech research at Microsoft," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, May 2013.

[4] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR) Workshop Track*, Banff, AB, Canada, April 2014.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the Int. Conf. On Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[6] Y. He, G. Meng, K. Chen, X. Hu, and J. He, "Towards privacy and security of deep learning systems: a survey," 2019, https://arxiv.org/abs/1911.12562v1.

[7] X. Yuan, He Pan, Q. Zhu, and X. Li, "Adversarial Examples: attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[8] K. Eykholt, I. Evtimov, E. Fernandes et al., "Robust physical-world attacks on deep learning models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Lake City, UT, USA, June 2018.

[9] Y. Dong, H. Su, B. Wu et al., "Efficient decision-based black-box Adversarial attacks on face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 2019.

[10] W. Fan, G. Sun, and X. Dong, "RGN-defense: erasing adversarial perturbations using deep residual generative network," *Journal of Electronic Imaging*, vol. 28, no. 1, 2019.

[11] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017.

[12] W. Fan, G. Sun, Y. Su, Z. Liu, and X. Lu, "Integration of statistical detector and Gaussian noise injection detector for adversarial example detection in deep neural networks," *Multimedia Tools and Applications*, vol. 78, no. 14, pp. 20409–20429, 2019.

[13] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "Imagenet: a large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, FL, USA, June 2009.

[14] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: attacks and defenses," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, April 2018.

[15] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proceedings of the International Conference on Learning Representations*, Toulon, France, April 2017.

[16] S. M. Moosavi Dezfooli, A. Fawzi, P. Frossard, and " Deepfool, "A simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016.

[17] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 39–57, San Jose, CA, USA, May 2017.

[18] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94, Honolulu, HI, USA, July 2017.

[19] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," 2015, https://arxiv.org/abs/1511.03034.

[20] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial

attacks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 2017.

[21] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *Computer Science*, vol. 14, no. 7, pp. 38-39, 2015.

[22] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural network," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 582–597, San Jose, CA, USA, May 2016.

[23] N. Papernot and P. McDaniel, "Extending defensive distillation," 2017, https://arxiv.org/abs/1705.05264.

[24] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," in *Proceedings of the International Conference. on Learning Representations (ICLR)*, pp. 1–9, San Diego, CA, USA, May 2015.

[25] G. K. Santhanam and P. Grnarova, "Defending against adversarial attacks by leveraging an entire GAN," 2018, https://arxiv.org/abs/1805.10652.

[26] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-GAN: Protecting classifiers against adversarial attacks using generative models," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, Canada, May, 2018.

[27] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, Canada, May, 2018.

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

[29] X. Jia, X. Wei, X. Cao, H. Foroosh, and " ComDefend, "An efficient image compression model to defend adversarial examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (CVPR)*, Long Beach, CA, USA, June 2019.

[30] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017, https://openreview.net/pdf?id=SJzCSf9xg.

[31] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," 2017, https://arxiv.org/abs/1702.06280.

[32] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," in *Proceedings of the International Conference on Machine Learning (ICML)*, Sydney, Australia, August 2017.

[33] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: detecting adversarial examples in deep neural networks," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2017.

[34] Y. Song, T. Kim, S. Nowozin, S. Ermon, N. Kushman, and " PixelDefend, "Leveraging generative models to understand and defend against adversarial examples," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, April 2018.

[35] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147, London, UK, November 2017.

[36] N. Akhtar, J. Liu, and A. Mian, "Defense against universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3389–3398, Salt Lake City, UT, USA, June 2018.

[37] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, 2010.

[38] J. Kodovsky, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.

[39] M. Abadi, "Tensorflow: a system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 265–283, Savannah, GA, USA, November 2016.

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.

[41] N. Papernot, F. Faghri, N. Carlini et al., "Technical report on the cleverhans v2.1.0 adversarial examples library," 2018, https://arxiv.org/abs/1610.00768v6.