Detection based Defense against Adversarial Examples from the Steganalysis Point of View

Jiayang Liu[1] Weiming Zhang[2] Yiwei Zhang[1] Dongdong Hou[1]
Yujia Liu[1] Hongyue Zha[1] Nenghai Yu[2]
University of Science and Technology of China
[1]{ljyljy,zywvvd,houdd,yjcaihon,zha00}@mail.ustc.edu.cn, [2]{zhangwm,ynh}@ustc.edu.cn

Abstract

Deep Neural Networks (DNNs) have recently led to significant improvements in many fields. However, DNNs are vulnerable to adversarial examples which are samples with imperceptible perturbations while dramatically misleading the DNNs. Moreover, adversarial examples can be used to perform an attack on various kinds of DNN based systems, even if the adversary has no access to the underlying model. Many defense methods have been proposed, such as obfuscating gradients of the networks or detecting adversarial examples. However it is proved out that these defense methods are not effective or cannot resist secondary adversarial attacks. In this paper, we point out that steganalysis can be applied to adversarial examples detection, and propose a method to enhance steganalysis features by estimating the probability of modifications caused by adversarial attacks. Experimental results show that the proposed method can accurately detect adversarial examples. Moreover, secondary adversarial attacks are hard to be directly performed to our method because our method is not based on a neural network but based on high-dimensional artificial features and Fisher Linear Discriminant ensemble.

1. Introduction

Deep Neural Networks (DNNs) have recently led to significant improvements in many fields, such as image classification [28, 15] and speech recognition [1]. However, the generalization properties of the DNNs have been recently questioned because these machine learning models are vulnerable to adversarial examples [31]. An adversarial example is a slightly modified sample that is intended to cause an error output of the DNN based model. In the context of classification task, the adversarial example is crafted to force a model to classify it into a class different from the legitimate class. In addition, adversarial examples have cross-model generalization property [12], so the attacker can even generate adversarial examples without the knowledge of the DNN. Adversarial attacks are divided into two types: targeted attack and untargeted attack. In targeted attack, the attacker generates adversarial examples which are misclassified by the classifier into a particular class. In untargeted attack, the attacker generates adversarial examples which are misclassified by the classifier into any class as long as it is different from the true class.

There are many studies which focus on methods of generating adversarial examples. Some attack methods are based on calculating the gradient of the network, such as Fast Gradient Sign Method (FGSM) [12], Basic Iterative Method (BIM) [18] and Jacobian Saliency Map Attack Method [25]. While other methods are based on solving optimization problems, such as L-BFGS [31], Deepfool [23] and Carlini & Wagner (C&W) attack [6].

Many defenses are proposed to mitigate adversarial examples against the above attacks. They make it harder for the adversary to craft adversarial examples using existing techniques or make the DNNs still give correct classifications on adversarial examples. These defenses are mainly divided into two categories.

One way is preprocessing the input image before classification, taking advantage of the spatial instability of adversarial examples. Defenders can perform some operations on the input image in spatial domain before giving the input image to a DNN, such as JPEG compression, scaling, adding noise, etc. Gu et al. [14] propose to use an autoencoder to remove adversarial perturbations from inputs.

The other way is to modify the network architecture, the optimization techniques or the training process. Goodfellow et al. [12] propose to augment the training set with adversarial examples to increase the model's robustness against a specific adversarial attack. However, this approach faces difficulties because the dimension of the images and features in networks means a great quantity of training data is required. Defensive distillation [26] is another technique against certain adversarial attacks. This form of network can prevent the model from fitting too tightly to the original

data. Unfortunately, most of these defenses are not very effective against adversarial examples in classification tasks.

Obfuscated gradients appear to be robust against adversarial attacks. Obfuscated gradients can be defined as a special case of gradient masking [24], in which the attackers cannot compute out the feasible gradient to generate adversarial examples. However, Athalye et al. [2] proposed attack techniques to overcome the obfuscated gradient based defenses.

Due to the difficulty of classifying adversarial examples correctly, recent work has turned to detecting them. Hendrycks & Gimpel [16], Li et al. [19] and Bhagoji et al. [3] use PCA to detect statistical properties of the images or network parameters. Feinman et al. [9] perform another statistical test to detect adversarial examples. Grosse et al. [13], Gong et al. [11] and Metzen et al. [22] utilize a second neural network to classify images as normal or adversarial. Lu et al. [21] detect adversarial examples by hypothesizing that adversarial examples produce different patterns of ReLU activations in networks than what is produced by normal images. Xu et al. [32] propose a method, called Feature Squeezing, to detect adversarial examples by measuring the disagreement among the prediction vectors of the original and squeezed examples. Liang et al. [20] detect the adversarial example by comparing the classification results of the input and its denoised version.

Unfortunately, Carlini and Wagner perform experiments to prove that most of these detecting methods are only effective on image datasets with small size or only several classes [5]. Moreover, Grosse's [13], Gong's [11] and Metzen's [22] defenses use a second neural network to classify images as normal or adversarial. However, neural networks used for detecting adversarial examples can also be bypassed [5]. In fact, given an adversarial method can fool the original neural network, Carlini et al. [5] show that with a similar method we can also fool the extended network for detection, which we call secondary adversarial attacks.

In this paper we propose to detect adversarial examples from the view of steganalysis [27] which is the technology for detecting steganography. In fact, Goodfellow et al. [12] have provided the insight on one essence of adversarial examples such that "the adversarial attack can be treated as a sort of accidental steganography". Furthermore, we propose a method to enhance steganalysis features by estimating the probability of modifications caused by adversarial attacks. Experimental results show that the proposed method can accurately detect adversarial examples. Moreover, secondary adversarial attacks are hard to be directly performed to our method because our method is not based on a neural network but based on high-dimensional artificial features and FLD (Fisher Linear Discriminant) ensemble.

2. Related Work

2.1. Adversarial Attacks

2.1.1 Fast Gradient Sign Method

Goodfellow et al. [12] propose the Fast Gradient Sign Method (FGSM) for generating adversarial examples. This method uses the derivative of the loss function of the network pertaining to the input feature vector. Given the input image X, FGSM is to perturb the gradient direction of each feature by the gradient. Then the classification result of the input image will be changed. For a neural network with cross-entropy cost function $J(X, y)$ where X is the input image and y_t is the target class for the input image, the adversarial example is generated as

$$X^{adv} = X - \epsilon \text{sign}(\nabla_X J(X, y_t)), \tag{1}$$

where ϵ is a parameter to determine the perturbation size.

2.1.2 Basic Iterative Method

The Basic Iterative Method (BIM) is the iterative version of FGSM. This method applies FGSM many times with small perturbation size instead of applying adversarial noise with one large perturbation size. The adversarial example of BIM is generated as

$$\begin{aligned} X_0^{adv} &= X, \\ X_{N+1}^{adv} &= Clip_{X,\epsilon}\{X_N^{adv} - \alpha \text{sign}(\nabla_X J(X_N^{adv}, y_t))\}, \end{aligned} \tag{2}$$

where $Clip_{X,\epsilon}\{X'\}$ represents a clipping of the pixel values after each iteration. So the results stay in the ϵ-neighbourhood of the input image X. This attack is more powerful because the attacker can control how far the adversarial example past the classification boundary. It was demonstrated that the attack of BIM was better than FGSM on ImageNet [18].

2.1.3 Deepfool

Deepfool is an untargeted attack method to generate an adversarial example by iteratively perturbing an image [23]. This method explores the nearest decision boundary. The image is modified a little to reach the boundary in each iteration. The algorithm stops once the modified image changes the classification of the network.

2.1.4 Carlini & Wagner Method

This method is named after its authors [6]. The attack can be targeted or untargeted, and has three metrics to measure its distortion (l_0 norm, l_2 norm and l_∞ norm). The authors point out that the untargeted l_2 norm version has the best

performance. It generates adversarial examples by solving the following optimization problem:

$$\begin{aligned} &\underset{\delta}{\text{minimize}} \, \|\delta\|_2 + c \cdot f(x+\delta) \\ &s.t. \quad x+\delta \in [0,1]^n \end{aligned} \tag{3}$$

This attack is to look for the smallest perturbation measured by l_2 norm and make the network classify the image incorrectly at the same time. c is a hyperparameter to balance the two parts of equation (3). The best way to choose c is to use the smallest value of c for which the resulting solution $x+\delta$ has $f(x+\delta) \leq 0$. $f(x)$ is the loss function to measure the distance between the input image and the adversarial image. $f(x)$ is defined as:

$$f(x) = \max(Z(x)_{true} - \max_{i \neq true}\{Z(x)_i\}, -\kappa). \tag{4}$$

$Z(x)$ is the pre-softmax classification result vector. κ is a hyper-parameter called confidence. Higher confidence encourages the attack to search for adversarial examples that are stronger in classification confidence. High-confidence attacks often have larger perturbations and better transferability to other models. The C&W method is a strong attack which is difficult to defend.

2.2. Robustness Based Defense

Robustness based defense aims at classifying adversarial examples correctly. There are many methods to achieve robustness based defense. Adversarial training is to train a better network by using a mixture of normal and adversarial examples in the training set for data augmentation [12]. Preprocessing the input images is to perform some operations to remove adversarial perturbations, such as principal component analysis (PCA) [3], JPEG compression [7], adding noise, cropping, rotating and so on. Defensive distillation hides the gradient between the pre-softmax layer and softmax outputs by leveraging distillation training techniques [26]. Obfuscated gradients make the attackers hard to compute out the feasible gradient to generate adversarial examples [2].

2.3. Detection Based Defense

Detection based defense aims at distinguishing normal images and adversarial examples.

Hendrycks & Gimpel [16] leverage PCA to detect adversarial examples, finding that adversarial examples place a higher weight on the larger principal components than normal images. However, Carlini and Wagner prove that Hendrycks's defense is only effective on MNIST [5].

Li et al. [19] apply PCA to the values after inner convolutional layers of the neural network, and use a cascade classifier to detect adversarial examples. Specifically, they propose building a cascade classifier that accepts the input as normal only if all classifiers accept the input, but rejects it if any do. However, Carlini and Wagner perform experiments to prove that Li's defense fails against the C&W attack [5].

Grosse et al. [13] propose a variant on adversarial retraining. Instead of attempting to classify the adversarial examples correctly, they introduce an additional class, solely for adversarial examples, and retrain the network to classify adversarial examples as the new class. Gong et al. [11] propose a very similar defense method. However, Carlini and Wagner re-implement these two defenses and find that they are only effective on MNIST [5].

Bhagoji et al. [3] leverage PCA to reduce the dimensionality of the images. Then instead of training on the original images, they train a classifier on images which have been processed with dimensionality reduction. However, this defense is only effective on MNIST [5].

Feinman et al. [9] utilize a Gaussian Mixture Model to model outputs from the final hidden layer of a neural network, and claim that adversarial examples belong to a different distribution than that of normal images. However, Carlini and Wagner prove that Feinman's defense is only effective on MNIST [5].

Metzen et al. [22] detect adversarial examples by looking at the inner convolutional layers of the network. They augment the classification neural network with a detection neural network that takes its input from various intermediate layers of the classification network. However, this defense is only effective on CIFAR-10 [5].

Lu et al. [21] hypothesize that adversarial examples produce different patterns of ReLU activations in networks than what is produced by normal images. Based on this hypothesis, they propose the Radial Basis Function SVM (RBF-SVM) classifier which takes advantage of discrete codes computed by the late stage ReLUs of the network to detect adversarial examples on CIFAR-10 and ImageNet.

Xu et al. [32] propose a method, called Feature Squeezing (FS), to detect adversarial examples. They reduce the color bit depth of each pixel and smooth it by a spatial filter to squeeze the features of an image. Then the adversarial examples are identified by measuring the disagreement among the prediction vectors of the original and squeezed examples.

Liang et al. [20] regard the adversarial perturbation as a kind of noise and use scalar quantization and smoothing spatial filter to reduce its adversarial effect. Then the adversarial example can be detected by comparing the classification results of the input and its denoised version. We refer to this method as Noise Reduction (NR).

For practical applications, we can deploy detection based defense combing with robustness based defense. First of all, we use detection based defense to detect the input image. If it is a normal image, we will directly feed it to the original

DNN. Otherwise we can take advantage of robustness based defense to mitigate adversarial examples.

3. Proposed Method

Both adversarial attacks and steganography on images make perturbations on the pixel values, which alter the dependence between pixels. However, steganalysis can effectively detect modifications caused by steganography via modeling the dependence between adjacent pixels in natural images. So we can also take advantage of steganalysis to identify deviations due to adversarial attacks.

Assuming that we have known the attacking method used by the attacker, we construct a detector to detect whether the input image is an adversarial example or not. In practice, we don't know the method used by the attacker, but we can deploy a series of detectors trained for various mainstream adversarial attacks. Our detection method exploits the fact that the perturbation of pixel values by adversarial attack alters the dependence between pixels. By modeling the differences between adjacent pixels in natural images, we can identify deviations due to adversarial attacks. In the beginning, we use a filter to suppress the content of the input image. Dependence between adjacent pixels of the filtered image is modeled as a higher order Markov chain [30]. Then the transition probability matrix is used as a vector feature for a feature based detector implemented using machine learning algorithms.

We recommend two kinds of steganalysis feature sets for detecting adversarial examples: one is the low-dimensional model SPAM with 686 features [27]; the other is the high-dimensional model Spatial Rich Model (SRM) with 34671 features [10].

3.1. Features Extraction

3.1.1 SPAM

SPAM is described as follows. First, we calculate the transition probabilities between pixels in eight directions $\{\leftarrow, \rightarrow, \downarrow, \uparrow, \nwarrow, \searrow, \swarrow, \nearrow\}$ in the spatial domain. We always compute the differences and the transition probability along the same direction. For example, the horizontal direction from left-to-right differences are calculated by $A^{\rightarrow}_{i,j} = X_{i,j} - X_{i,j+1}$, where X is an image with size of $m \times n$, and $X_{i,j}$ is the pixel at the position (i, j) for $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n-1\}$. Second, we use a Markov chain between pairs of differences (first order chain) or triplets (second order chain) to model pixel dependence along the eight directions. The first-order detecting features, F^{1st}, model the difference arrays A via a first-order Markov process. For the horizontal direction, this leads to

$$M^{\rightarrow}_{x,y} = Pr(A^{\rightarrow}_{i,j+1} = x | A^{\rightarrow}_{i,j} = y) \tag{5}$$

where $x, y \in \{-T, \ldots, T\}$. The second-order detecting features, F^{2nd}, model the difference arrays A via a second-order Markov process. For the horizontal direction, this leads to

$$M^{\rightarrow}_{x,y,z} = P(A^{\rightarrow}_{i,j+2} = x | A^{\rightarrow}_{i,j+1} = y, A^{\rightarrow}_{i,j} = z) \tag{6}$$

where $x, y, z \in \{-T, \ldots, T\}$. To reduce the dimensionality of the transition probability matrix, we only consider differences within a limited range. Thus, we just calculate the transition probability matrix for pairs within $[-T, T]$. We separately average the horizontal and vertical matrices and then the diagonal matrices to form the final feature sets, F^{1st}, F^{2nd}. The expression of the average sample Markov transition probability matrices is

$$\begin{aligned} F_{1,\ldots,k} &= (M^{\rightarrow} + M^{\leftarrow} + M^{\uparrow} + M^{\downarrow})/4 \\ F_{k+1,\ldots,2k} &= (M^{\nearrow} + M^{\swarrow} + M^{\searrow} + M^{\nwarrow})/4 \end{aligned} \tag{7}$$

where $k = (2T+1)^2$ for the first-order detecting features and $k = (2T+1)^3$ for the second-order detecting features. We can see that the order of Markov model and the range of differences T control the dimensionality of our detecting model. We use $T = 3$ for second order, resulting in $2k = 686$ features [27].

3.1.2 Spatial Rich Model

Spatial Rich Model (SRM) can be viewed as an extended version of SPAM by extracting residuals from images [10]. We use a pixel predictor from the pixel's immediate neighborhood to obtain a residual which is an estimate of the image noise component. SRM uses 45 different pixel predictors. The pixel predictor is linear or non-linear. Each linear predictor is a shift-invariant finite-impulse response filter which is described by a kernel matrix $K^{(pred)}$. The residual $Z = (z_{kl})$ is a matrix which has the same dimension as X:

$$Z = K^{(pred)} * X - X \triangleq K * X, \tag{8}$$

where the symbol $*$ denotes the convolution with X mirror-padded. Thus, $K * X$ has the same dimension as X.

For example, one simple linear residual is $z_{ij} = X_{i,j+1} - X_{i,j}$, which is the difference between a pair of horizontally adjacent pixels. In this case, the residual kernel is $K = \begin{pmatrix} -1 & 1 \end{pmatrix}$, which means that the pixel value is predicted as its horizontally neighboring pixel.

SRM obtains non-linear predictors by taking the minimum or maximum of up to five residuals which are obtained by using linear predictors. For example, we can predict pixel $X_{i,j}$ from its horizontal or vertical neighboring pixels, obtaining thus one horizontal and one vertical residual $Z^{(h)} = \left(z^{h}_{ij}\right)$, $Z^{(v)} = \left(z^{v}_{ij}\right)$:

$$z^{(h)}_{ij} = X_{i,j+1} - X_{i,j}, \tag{9}$$

$$z_{ij}^{(v)} = X_{i+1,j} - X_{i,j}. \tag{10}$$

We can use these two residuals to compute two nonlinear "minmax" residuals as:

$$z_{ij}^{(\min)} = \min\left\{z_{ij}^{(h)}, z_{ij}^{(v)}\right\}, \tag{11}$$

$$z_{ij}^{(\max)} = \max\left\{z_{ij}^{(h)}, z_{ij}^{(v)}\right\}. \tag{12}$$

After that, quantize Z with a quantizer $Q_{-T,T}$ with centroids $Q_{-T,T} = \{-Tq, (-T+1)\,q, \ldots, Tq\}$, where $T > 0$ is an integer threshold and $q > 0$ is a quantization step:

$$r_{ij} \triangleq Q_{-T,T}\left(z_{ij}\right), \forall i, j. \tag{13}$$

The next step is that a co-occurrence matrix of fourth order, $C^{(SRM)} \in Q_{-T,T}^4$, is computed from four (horizontally and vertically) adjacent values of the quantized residual r_{ij} from the entire image:

$$C_{d_0 d_1 d_2 d_3}^{SRM} = \sum_{i,j=1}^{m,n-3} [r_{i,j} = d_k, \forall k = 0, \ldots, 3], \tag{14}$$

where $d_k \in Q_{-T,T}$ and $[B]$ is the Iverson bracket, which is 1 if the statement B is true and 0 otherwise. The union of all co-occurrence matrices has a total dimension of 34671.

3.2. Features Enhancement

The above methods of extracting steganalysis features do not consider the location of modified pixels caused by adversarial attacks. Obviously, the detection rate will be improved if we assign larger weight to the features of the modified location. Although we cannot obtain the accurate modified location, we can estimate the relative modification probability of each pixel. In order to further improve the accuracy of detection, we propose to enhance steganalysis features by estimating the probability of modifications caused by adversarial attacks.

We take advantage of the gradient amplitude to estimate the modification probability because the pixels with larger gradient amplitude have larger probability to be modified. Assume that the neural network divides images into N categories. Although we cannot know which target class will be selected by the attacker, we can randomly select L categories to generate L targeted adversarial examples and then estimate the modification probability of each pixel according to these targeted adversarial examples. So we take the t-th $(1 \leq t \leq L)$ class as the target class to calculate the gradient of the input image X. We refer to the matrix of all pixels' modification probabilities as Modification Probability Map (MPM). Note that these targeted adversarial examples generated by us are only used to estimate MPM which can be used to enhance the detection of adversarial attacks.

For FGSM and BIM, when generating the adversarial example of the target class y_t for the input image, we save absolute values of the gradient of each pixel $|\nabla_X J(X, y_t)|$, and then normalize them to obtain the gradient map $f_{nor}\left(|\nabla_X J(X, y_t)|\right)$ where $f_{nor}\left(\right)$ is the function to normalize all elements in the matrix to $(0, 1)$. Finally, calculate the mean value of the gradient maps of L adversarial examples to get MPM P:

$$P = \frac{1}{L}\sum_{t=1}^{L} f_{nor}\left(|\nabla_X J(X, y_t)|\right), \tag{15}$$

where P is a $m \times n$ matrix in which the element $P_{i,j}$ is the modification probability of the pixel $X_{i,j}$.

For C&W which does not generate adversarial examples by gradient, the estimation of MPM starts by computing the difference array D_t between the normal image X and the adversarial example X_t^{adv}:

$$D_t = X_t^{adv} - X. \tag{16}$$

Then save the absolute values of all elements in the difference array D_t and normalize them to obtain the difference map $f_{nor}\left(|D_t|\right)$. Finally, calculate the mean value of the difference maps of L adversarial examples to get MPM:

$$P = \frac{1}{L}\sum_{t=1}^{L} f_{nor}\left(|D_t|\right). \tag{17}$$

For Deepfool which can only generate untargeted adversarial examples, we estimate MPM by computing the difference array D between the normal image X and the adversarial example X^{adv}:

$$D = X^{adv} - X. \tag{18}$$

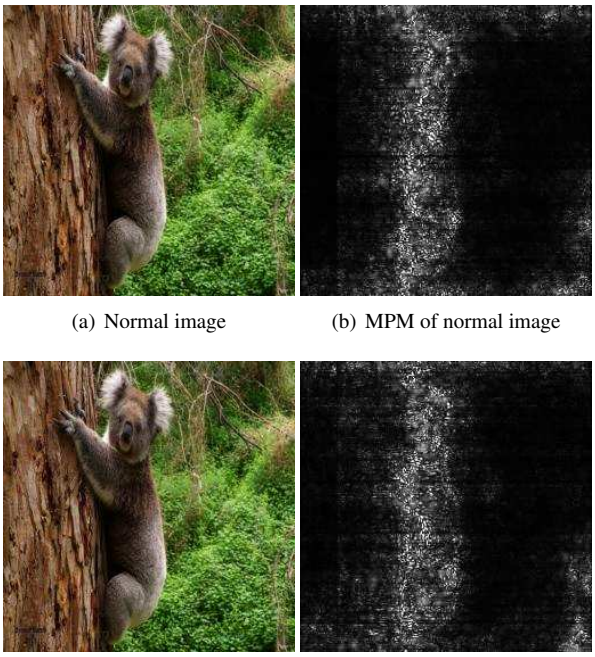
Then save the absolute values of all elements in the difference array D and normalize them to obtain MPM:

$$P = f_{nor}\left(|D|\right). \tag{19}$$

The above description is the estimation of MPM based on normal images. In practice, the detector may receive an adversarial example. The results of our experiments show that the MPM of one normal image and its adversarial image is quite similar. Figure 1 shows an example of a normal image, an adversarial image and their MPM (normalized to $(0, 255)$ to show more clearly).

3.2.1 Enhanced SPAM

Considering the impact of MPM, Enhanced SPAM (ESPAM) is proposed. The difference between SPAM and ESPAM is that we construct a new Markov transition probability based on MPM. For example, in the horizontal direction, the Markov transition probability $M_{x,y}^{\rightarrow}$ is related to



(a) Normal image (b) MPM of normal image

(c) Adversarial image (d) MPM of adversarial image

Figure 1. Illustrations of a normal image, an adversarial image and their MPM.

the pixel $X_{i,j}$, $X_{i,j+1}$ and $X_{i,j+2}$. So we calculate the new Markov transition probability $M'^{\rightarrow}_{x,y}$ in this way:

$$M'^{\rightarrow}_{x,y} = M^{\rightarrow}_{x,y} \cdot P_{i,j} \cdot P_{i,j+1} \cdot P_{i,j+2}. \tag{20}$$

Similarly, for the second-order detecting features, the new Markov transition probability $M'^{\rightarrow}_{x,y,z}$ is

$$M'^{\rightarrow}_{x,y,z} = M^{\rightarrow}_{x,y,z} \cdot P_{i,j} \cdot P_{i,j+1} \cdot P_{i,j+2} \cdot P_{i,j+3}. \tag{21}$$

The rest of the process of forming ESPAM is the same with SPAM. ESPAM has the same dimensionality as SPAM, which is 686.

3.2.2 Enhanced SRM

The Enhanced Spatial Rich Model (ESRM) is built in the similar manner as SRM. The difference is that ESRM modifies the process of forming the co-occurrence matrices to consider the impact of MPM:

$$C^{ESRM}_{d_0 d_1 d_2 d_3} = \sum_{i,j=1}^{m,n-3} \max_{k=0,\ldots,3} P_{i,j+k} \left[r_{i,j} = d_k, \forall k = 0, \ldots, 3\right], \tag{22}$$

where $C^{(ESRM)}$ is the enhanced version of the co-occurrence $C^{(SRM)}$. In other words, instead of increasing the corresponding co-occurrence bin by 1, we add the maximum of the modification probabilities taken across the four residuals to the bin [8]. Thus, if a group has four pixels with small modification probabilities, it has smaller effect on the co-occurrence values than the group with at least one pixel likely to be changed. The rest of the process of forming ESRM stays exactly the same with SRM. ESRM has the same dimensionality as SRM, which is 34671.

3.3. Training Detector

The construction of our detectors based on features relies on pattern-recognition classifiers. The detectors are trained as binary classifiers implemented using the FLD (Fisher Linear Discriminant) ensemble [17] with default settings. The ensemble by default minimizes the total classification error probability under equal priors. We find the number of base learners and the random subspace dimensionality via minimizing the out-of-bag estimate of the testing error calculated from the training set because it is an unbiased estimate of the testing error on unseen data [4].

4. Experimental Results

We construct the detectors by modeling the statistical differences between adjacent pixels in natural images. Therefore, our method can not achieve very good performance on MNIST and CIFAR-10 because images with small sizes cannot provide enough samples to construct efficient features. However, it has good performance on ImageNet. Previous work showed that untargeted attack is easier to succeed, results in smaller perturbations, and transfers better to different models. So we detect untargeted adversarial examples to see the performance of our method.

We test our detection method against untargeted attacks by FGSM, BIM, Deepfool and C&W. Our experiments are performed on 40000 images randomly selected from ImageNet (ILSVRC-2016) using a pretrained VGG-16 model [29] as classification network which is evaluated with top-1 accuracy. This results in a train set of 25000 images, a validation set of 5000 images, and a test set of 10000 images. The values of pixels per color channel of these 40000 images range from 0 to 255. For BIM, we use $\alpha = 1$ to ensure that we change each pixel by 1 on each step and $\epsilon \leq 8$ where ϵ is a parameter to determine the perturbation size. For Deepfool, we apply the l_2 norm version. For C&W, we use the l_2 norm version and set $\kappa = 0$. In the process of estimating MPM, we set $L = 100$.

At first, the 40000 images from ImageNet are classified by the network to obtain their true labels. Then we use these 40000 images to generate 40000 adversarial images as adversarial samples of our experiments. To prove that MPM is effective when detecting adversarial examples, we perform comparative experiments. We construct two pairs of detectors: SPAM and ESPAM, SRM and ESRM. The only difference between each pair is one detector with MPM and the other detector without MPM. All detectors are trained and tested on the same adversarial method.

Table 1. Preliminary evaluations on previous detection methods.

detection method	effective on ImageNet	effective against C&W	without second neural network
Hendrycks's [16]	✗	✗	✓
Li's [19]	✓	✗	✓
Grosse's [13]	✗	✓	✗
Gong's [11]	✗	✓	✗
Bhagoji's [3]	✗	✗	✓
Feinman's [9]	✗	✗	✓
Metzen's [22]	✗	✗	✗
RBF-SVM [21]	✓	unknown	✓
FS [32]	✓	✓	✓
NR [20]	✓	✓	✓

Table 2. Detection rate of normal images and their adversarial images generated by FGSM.

RBF-SVM [21]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.8340	0.8913	0.9305	0.9487
adversarial images	0.8258	0.8936	0.9243	0.9541
FS [32]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9460	0.9472	0.9441	0.9455
adversarial images	0.4029	0.2856	0.2078	0.1715
NR [20]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.8774	0.8670	0.8538	0.8513
adversarial images	0.7752	0.6908	0.6324	0.5587
SPAM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9488	0.9570	0.9651	0.9713
adversarial images	0.9432	0.9559	0.9628	0.9709
ESPAM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9725	0.9758	0.9812	0.9868
adversarial images	0.9704	0.9719	0.9751	0.9806
SRM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9757	0.9814	0.9831	0.9887
adversarial images	0.9785	0.9822	0.9861	0.9903
ESRM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9809	0.9839	0.9900	0.9931
adversarial images	0.9811	0.9866	0.9905	0.9938

Carlini and Wagner point out that it is necessary to evaluate defenses using a strong attack on harder datasets (such as ImageNet) [5]. Moreover, Carlini and Wagner prove that using a second neural network to identify adversarial examples is the least effective defense [5]. Therefore we only compare our method with the defense which is effective against C&W on ImageNet and not based on another neural network. Table 1 illustrates preliminary evaluations on previous detection methods: whether effective on ImageNet, whether effective against C&W and whether without second neural network. As shown in Table 1, Li's defense [19] fails against the C&W attack. Hendrycks's [16], Bhagoji's [3] and Feinman's [9] defenses are only effective on MNIST.

Table 3. Detection rate of normal images and their adversarial images generated by BIM.

RBF-SVM [21]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.7749	0.8660	0.9145	0.9362
adversarial images	0.7975	0.8752	0.9072	0.9330
FS [32]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9457	0.9440	0.9466	0.9451
adversarial images	0.6281	0.3547	0.2592	0.2134
NR [20]	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.8802	0.8742	0.8599	0.8530
adversarial images	0.8210	0.7411	0.6725	0.6143
SPAM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9402	0.9485	0.9559	0.9606
adversarial images	0.9411	0.9474	0.9545	0.9601
ESPAM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9708	0.9737	0.9749	0.9760
adversarial images	0.9638	0.9675	0.9725	0.9745
SRM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9667	0.9706	0.9753	0.9802
adversarial images	0.9697	0.9724	0.9762	0.9812
ESRM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.9712	0.9754	0.9811	0.9878
adversarial images	0.9716	0.9767	0.9820	0.9879

Table 4. Detection rate of normal images and their adversarial images generated by Deepfool.

	normal images	adversarial images
RBF-SVM [21]	0.5838	0.6012
FS [32]	0.9476	0.7441
NR [20]	0.9021	0.9208
SPAM	0.8553	0.8481
ESPAM	0.8690	0.8572
SRM	0.9445	0.9491
ESRM	0.9498	0.9527

Table 5. Detection rate of normal images and their adversarial images generated by C&W.

	normal images	adversarial images
RBF-SVM [21]	0.5332	0.5187
FS [32]	0.9485	0.8933
NR [20]	0.9110	0.9226
SPAM	0.6957	0.6778
ESPAM	0.7467	0.7563
SRM	0.8814	0.9092
ESRM	0.9233	0.9341

Table 6. Detection rate of secondary adversarial attacks yielded by C&W.

	SPAM	ESPAM	SRM	ESRM
adversarial images	0.6669	0.7246	0.8944	0.9150

Grosse's [13], Gong's [11] and Metzen's [22] defenses use

a second neural network to classify images as normal or adversarial. RBF-SVM [21] has good performance on ImageNet even though its performance against C&W is not evaluated. FS [32] and NR [20] claim good performance when detecting C&W. Therefore, we compare our detectors with RBF-SVM, FS and NR.

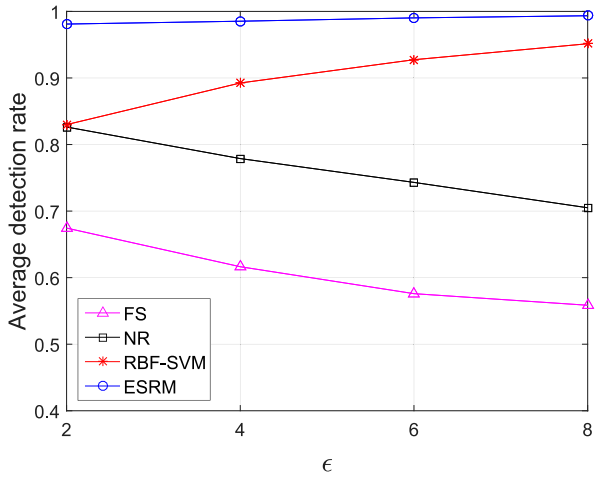


Figure 2. Average detection rate for detectors against FGSM.

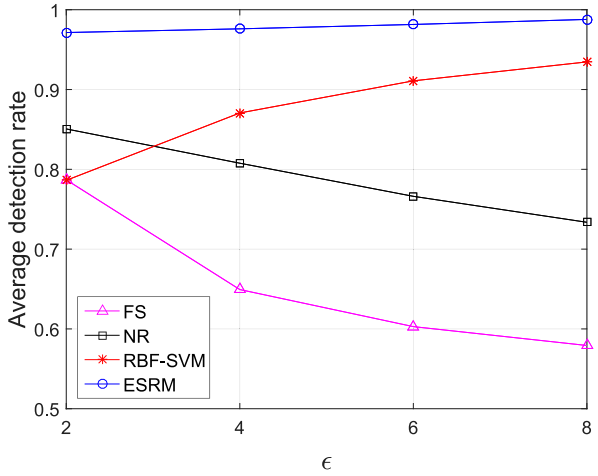


Figure 3. Average detection rate for detectors against BIM.

The experimental results of detecting adversarial examples are shown in Table 2,3,4,5. The data of Table 2,3,4,5 is the detection rate of normal images and adversarial images. Figure 2 and Figure 3 illustrate these detectors' performance by averaging the detection rate of detecting normal images and adversarial images. First of all, the results reveal that the detectors with MPM have higher detection rates. Moreover, MPM has stronger enhancing effect on SPAM than SRM. When detecting FGSM and BIM, ESPAM even has comparable performance as SRM. That is to say, we can even use the low-dimensional model to achieve comparable performance as the high-dimensional model via the enhancing method. Experimental results show that it is difficult to detect adversarial examples generated by the C&W method. RBF-SVM is almost invalid against C&W. SPAM and SRM achieve relatively low detection rate when detecting C&W. However, MPM improves SPAM by more than 7 percent and the detection rate of ESRM reaches 93 percent on detecting adversarial examples yielded by C&W. FS and NR achieve comparable performance with ESRM against C&W. Unfortunately, the detection rates of FS and NR are much lower against FGSM and BIM. We suspect the reason why FS and NR are less effective against FGSM and BIM is that FS and NR are only well suited to mitigating small adversarial perturbations. On the contrary, the detection rate of ESRM is the highest on detecting adversarial examples generated by FGSM, BIM, Deepfool and C&W. However, the computation time of SRM and ESRM is much longer because of their high-dimensional features.

5. Secondary Adversarial Attacks

The secondary adversarial attacks are hard to be directly performed to our method because the structure of our detection model is not a neural network. A direct idea of escaping the detection of our method is to reduce the number of adversarial perturbations. However, this strategy will also weaken the ability of adversarial examples to mislead the DNN. To verify this point, we try to perform the secondary attacks by removing 10% of the untargeted adversarial perturbations generated by C&W. As shown in Table 6, the detection rate of ESRM drops from 93.41% to 91.50%. However, the success rate of secondary attack adversarial examples to deceive the network drops from 99.03% to 45.27%.

6. Conclusions

Inspired by the insight of Goodfellow et al. [12] that "adversarial examples can be thought of as a sort of accidental steganography", we propose to apply steganalysis to detecting adversarial examples. We also propose a method to enhance steganalysis features. The experimental results show that the enhanced scheme can accurately detect various kinds of adversarial attacks including the C&W method. Moreover, the secondary adversarial attacks [5] are hard to be directly performed to our method because the structure of our detection model is not a neural network. Our method draws a relevant connection between adversarial examples of computer vision and steganalysis and could kindle more promising work in this direction. As an idea for future work, a better secondary attack could try to add perturbations that maintain the dependence between neighboring pixels.

Acknowledgement This work was supported in part by the Natural Science Foundation of China under Grant U1636201 and 61572452, and by Anhui Initiative in Quantum Information Technologies under Grant AHY150400.

References

[1] Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[3] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. Dimensionality reduction as a defense against evasion attacks on machine learning classifiers. *arXiv preprint arXiv:1704.02654*, 2017.

[4] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[5] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017.

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy*, 2017.

[7] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

[8] Tomas Denemark, Vahid Sedighi, Vojtech Holub, Rémi Cogranne, and Jessica Fridrich. Selection-channel-aware rich model for steganalysis of digital images. In *Information Forensics and Security (WIFS), 2014 IEEE International Workshop on*, pages 48–53. IEEE, 2014.

[9] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[10] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[11] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017.

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[13] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[14] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *International Conference on Learning Representations*, 2015.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[16] Dan Hendrycks and Kevin Gimpel. Early methods for detecting adversarial images. *arXiv preprint arXiv:1608.00530*, 2016.

[17] Jan Kodovsky, Jessica Fridrich, and Vojtěch Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2012.

[18] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *International Conference on Learning Representations*, 2017.

[19] Xin Li and Fuxin Li. Adversarial examples detection in deep networks with convolutional filter statistics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5764–5772, 2017.

[20] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and XiaoFeng Wang. Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Transactions on Dependable and Secure Computing*, 2018.

[21] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *IEEE International Conference on Computer Vision*, pages 446–454, 2017.

[22] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *International Conference on Learning Representations*, 2017.

[23] Seyed Mohsen Moosavidezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *Computer Vision and Pattern Recognition*, pages 2574–2582, 2015.

[24] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[25] Nicolas Papernot, Patrick Mcdaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy*, pages 372–387, 2016.

[26] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 582–597. IEEE, 2016.

[27] Tomáš Pevny, Patrick Bas, and Jessica Fridrich. Steganalysis by subtractive pixel adjacency matrix. *IEEE Transactions on information Forensics and Security*, 5(2):215–224, 2010.

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

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Computer Science*, 2014.

[30] Kenneth Sullivan, Upamanyu Madhow, Shivkumar Chandrasekaran, and Bangalore S Manjunath. Steganalysis of

spread spectrum data hiding exploiting cover memory. In *Electronic Imaging 2005*, pages 38–46. International Society for Optics and Photonics, 2005.

[31] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations*, 2014.

[32] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Proceedings of the 2018 Network and Distributed Systems Security Symposium (NDSS)*, 2018.