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

Jiayang Liu, Weiming Zhang, Yiwei Zhang, Dongdong Hou, Yujia Liu, Hongyue Zha and Nenghai Yu

CAS Key Laboratory of Electromagnetic Space Information
University of Science and Technology of China
Hefei, China

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 cannot 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.

Introduction

Deep Neural Networks (DNNs) have recently led to significant improvements in many fields, such as image classification (Russakovsky et al. 2015; He et al. 2016) and speech recognition (Amodei et al. 2016). However, the generalization properties of the DNNs have been recently questioned because these machine learning models are vulnerable to adversarial examples (Szegedy et al. 2014). 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 (Goodfellow, Shlens, and Szegedy 2015), 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 par-

ticular 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) (Goodfellow, Shlens, and Szegedy 2015), Iterative Gradient Sign Method (IGSM) (Kurakin, Goodfellow, and Bengio 2017) and Jacobian Saliency Map Attack Method (Papernot et al. 2016a). While other methods are based on solving optimization problems, such as L-BFGS (Szegedy et al. 2014), Deepfool (Moosavidezfooli, Fawzi, and Frossard 2015) and Carlini & Wagner (C&W) attack (Carlini and Wagner 2017b).

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. (Gu and Rigazio 2015) 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. (Goodfellow, Shlens, and Szegedy 2015) 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 an unreasonable quantity of training data is required. Zheng et al. (Zheng et al. 2016) propose to append a stability term to the objective function to force the model to have similar outputs for normal images of the training set and their adversarial examples. This is different from data augmentation because it encourages the smoothness of the model output between original and adversarial samples. Defensive distillation (Papernot et al. 2016b) is another technique against certain adversarial attacks. This form of network can pre-

vent 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 (Papernot et al. 2017), in which the attackers cannot compute out the feasible gradient to generate adversarial examples. However, Athalye et al. (Athalye, Carlini, and Wagner 2018) 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 (Hendrycks and Gimpel 2016) use PCA to detect natural images from adversarial examples, finding that adversarial examples place a higher weight on the larger principal components than normal images. Li et al. (Li and Li 2017) apply PCA to the values after inner convolutional layers of the neural network, and use a cascade classifier to detect adversarial examples. Grosse et al. (Grosse et al. 2017) propose a variant on adversarial re-training. They introduce a new class in the model solely for the adversarial examples, and train the network to detect adversarial examples. Gong et al. (Gong, Wang, and Ku 2017) construct a very similar defense technique as Grosse's defense. Metzen et al. (Metzen et al. 2017) propose to add a detection subnetwork which observes the state of the original classification network, one can tell whether it has been presented with an adversarial example or not. Lu et al. (Lu, Issaranon, and Forsyth 2017) detect adversarial examples by hypothesizing that adversarial examples produce different patterns of ReLU activations in networks than what is produced by normal images.

Unfortunately, Carlini & Wagner perform experiments to prove that most of these detecting methods are only effective on image databases with small size or only several classes. Moreover, Grosse's, Gong's and Metzen's 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 (Carlini and Wagner 2017a). In fact, given an adversarial method can fool the original neural network, Carlini et al. 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 (Pevny, Bas, and Fridrich 2010) which is the technology for detecting steganography. In fact, Goodfellow et al. (Goodfellow, Shlens, and Szegedy 2015) 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 cannot 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.

Related Work

Adversarial Attacks

Fast Gradient Sign Method   Goodfellow et al. (Goodfellow, Shlens, and Szegedy 2015) propose the Fast Gradient Sign Method (FGSM) for generating adversarial examples. This method uses the derivative of the loss function of the model 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.

Iterative Gradient Sign Method   The Iterative Gradient Sign Method (IGSM) 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 the iterative gradient sign method is generated as

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

where $Clip_{X,\epsilon}\{X'\}$ represents a clipping of the values of the adversarial example. So the results are within 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 IGSM was better than FGSM on ImageNet-1000 (Kurakin, Goodfellow, and Bengio 2017).

Deepfool   Deepfool is an untargeted attack method to generate an adversarial example by iteratively perturbing an image (Moosavidezfooli, Fawzi, and Frossard 2015). 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.

Carlini & Wagner Method   This method is named after its authors (Carlini and Wagner 2017b). 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:

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

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:

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

$Z(x)$ 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.

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, which we refer to as adversarial training (Goodfellow, Shlens, and Szegedy 2015). Preprocessing the input images is to perform some operations to remove adversarial perturbations, such as principal component analysis (PCA) (Bhagoji, Cullina, and Mittal 2017), JPEG compression (Das et al. 2017), 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 (Papernot et al. 2016b). Obfuscated gradients make the attackers hard to compute out the feasible gradient to generate adversarial examples (Athalye, Carlini, and Wagner 2018).

Detection Based Defense

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

Hendrycks & Gimpel (Hendrycks and Gimpel 2016) use PCA to detect natural images from adversarial examples, finding that adversarial examples place a higher weight on the larger principal components than normal images. However, the Hendrycks defense is only effective for MNIST.

Li et al. (Li and Li 2017) 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 natural only if all classifiers accept the input, but rejects it if any do. However, Carlini & Wagner perform experiments to prove that Li's defense fails against the C&W attack (Carlini and Wagner 2017a).

Grosse et al. (Grosse et al. 2017) propose a variant on adversarial re-training. Instead of attempting to classify the adversarial examples correctly, they introduce the $(N+1)\,th$ class, solely for adversarial examples, and train the network to detect adversarial examples. Gong et al. (Gong, Wang, and Ku 2017) construct a very similar defense technique. However, Carlini & Wagner re-implement these two defenses and find that they are only effective for MNIST (Carlini and Wagner 2017a).

Metzen et al. (Metzen et al. 2017) 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 against CIFAR-10.

Lu et al. (Lu, Issaranon, and Forsyth 2017) 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-1000.

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.

Proposed Method

Both adversarial attacks and steganography make perturbations on the pixel values, which alter the dependence between pixels. However, steganalysis can effectively detect modifications caused by steganography by 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 (Sullivan et al. 2005). 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 (Pevny, Bas, and Fridrich 2010); the other is the high-dimensional model Spatial Rich Model (SRM) with 34671 features (Fridrich and Kodovsky 2012).

Features Extraction

SPAM SPAM is described as follows. First, the model calculates the transition probabilities between pixels in eight directions $\{\leftarrow, \rightarrow, \downarrow, \uparrow, \nwarrow, \searrow, \swarrow, \nearrow\}$ in the spatial domain. The differences and the transition probability are always computed along the same direction. For example, the horizontal direction from left-to-right differences are calculated

by $A_{i,j}^{\rightarrow} = X_{i,j} - X_{i,j+1}$, 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, to model pixel dependence along the eight directions, a Markov chain is used between pairs of differences (first order chain) or triplets (second order chain). The first-order detecting features, F^{1st}, model the difference arrays A by a first-order Markov process. For the horizontal direction, this leads to

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

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

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

where $x, y, z \in \{-T, \ldots, T\}$. For dimensionality reduction of the transition probability matrix, only differences within a limited range are considered. Thus, the transition probability matrix is calculated just 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 (Pevny, Bas, and Fridrich 2010).

Spatial Rich Model Spatial Rich Model (SRM) can be viewed as an extended version of SPAM by extracting residuals from images (Fridrich and Kodovsky 2012). A residual is an estimate of the image noise component obtained by subtracting from each pixel its estimate obtained using a pixel predictor from the pixel's immediate neighborhood. SRM uses 45 different pixel predictors of two different types: linear and non-linear. Each linear predictor is a shift-invariant finite-impulse response filter described by a kernel matrix $K^{(pred)}$. The residual $Z = (z_{kl})$ is a matrix of 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 so that $K * X$ has the same dimension as X.

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

All non-linear predictors in the SRM are obtained by taking the minimum or maximum of up to five residuals obtained using linear predictors. For example, one can predict

pixel $X_{i,j}$ from its horizontal or vertical neighbors, obtaining thus one horizontal and one vertical residual $Z^{(h)} = (z_{ij}^h)$, $Z^{(v)} = (z_{ij}^v)$:

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

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

Using these two residuals, one can 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:

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

The next step in forming the SRM feature vector involves computing a co-occurrence matrix of fourth order, $C^{(SRM)} \in Q_{-T,T}^4$, from four (horizontally and vertically) neighboring 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 $[B]$ is the Iverson bracket, which is equal to 1 when the statement B is true and to 0 when it is false. The union of all co-occurrence matrices, including their differently quantized versions, has a total dimension of 34671.

Features Enhancement

The above methods of extracting steganalysis features do not consider the location of modified pixels caused by adversarial attacks. Obviously, the accuracy of detection will be improved if we assign larger weight to the features of 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 ith $(1 \leq i \leq L)$ 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 detect untargeted attacks.

For FGSM and IGSM, when generating the adversarial example of the target class y_i for the input image, we save absolute values of the gradient of each pixel $|\nabla_X J(X, y_i)|$, and then normalize them to obtain the gradient map $f_{nor}(|\nabla_X J(X, y_i)|)$ where $f_{nor}()$ 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:

$$P = \frac{1}{L} \sum_{i=1}^{L} f_{nor}(|\nabla_X J(X, y_i)|) \tag{15}$$

P is a $m \times n$ matrix in which the $(i, j)th$ element $P_{i,j}$ is the modification probability of the $(i, j)th$ 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_i between the normal image X and the adversarial example X_i^{adv}:

$$D_i = X_i^{adv} - X \tag{16}$$

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

$$P = \frac{1}{L} \sum_{i=1}^{L} f_{nor}(|D_i|) \tag{17}$$

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:

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

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).

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

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

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

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

Then the expression of the average sample Markov transition probability matrices is

$$F_{1,...,k} = (M'^{\rightarrow} + M'^{\leftarrow} + M'^{\uparrow} + M'^{\downarrow})/4$$
$$F_{k+1,...,2k} = (M'^{\nearrow} + M'^{\swarrow} + M'^{\searrow} + M'^{\nwarrow})/4 \tag{22}$$

where $k = (2T + 1)^2$ for the first-order detecting features and $k = (2T + 1)^3$ for the second-order detecting features. ESPAM has the same dimensionality as SPAM, which is 686.

Enhanced SRM The Enhanced Spatial Rich Model (ESRM) is built in the same manner as the SRM but the process of forming the co-occurrence matrices is modified to consider the impact of MPM:

$$C_{d_0 d_1 d_2 d_3}^{ESRM} = \sum_{i,j=1}^{m,n-3} \max_{k=0,...,3} P_{i,j+k} [r_{i,j} = d_k, \forall k = 0, \ldots, 3] \tag{23}$$

Above, $C^{(ESRM)}$ denotes the enhanced version of the co-occurrence $C^{(SRM)}$. In other words, instead of increasing the corresponding co-occurrence bin by 1, the maximum of the modification probabilities taken across the four residuals is added to the bin (Denemark et al. 2014). 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.

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 ensemble (Kodovsky, Fridrich, and Holub 2012) with default settings. The ensemble by default minimizes the total classification error probability under equal priors. The random subspace dimensionality and the number of base learners is found by minimizing the out-of-bag estimate of the testing error on bootstrap samples of the training set as it is an unbiased estimate of the testing error on unseen data (Breiman 1996).

Experimental Results

We construct the detectors by modeling the differences between adjacent pixels in natural images. Therefore, our method can not achieve very good performance on MNIST and CIFAR-10 because the size of the image is too small. However, it has good performance on ImageNet-1000. 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 detecting method against untargeted attacks by FGSM, IGSM, Deepfool and C&W. Our experiments are performed on 40000 images randomly selected from ImageNet-1000 (ILSVRC-2016) using a pretrained VGG-16 model (Simonyan and Zisserman 2014) as classification network which is evaluated with top-1 accuracy. This results

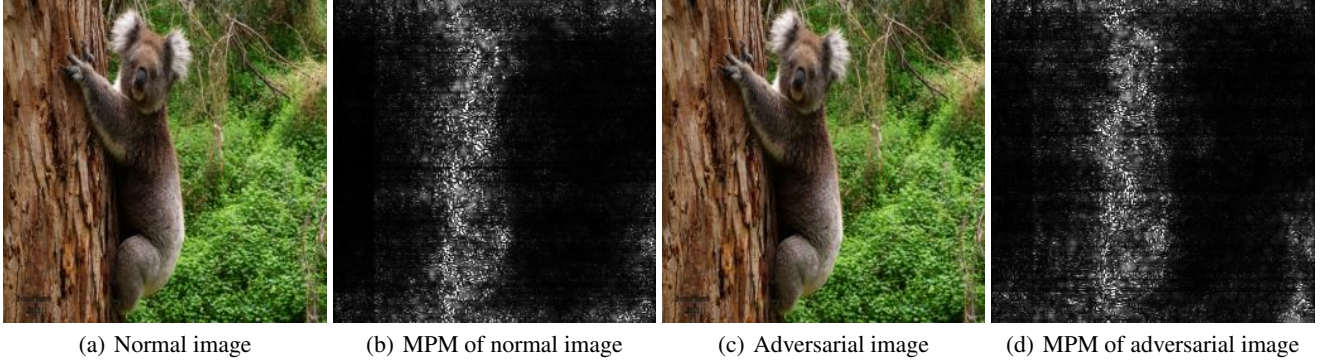| (a) Normal image | (b) MPM of normal image | (c) Adversarial image | (d) MPM of adversarial image |

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

Table 1: Detection accuracy of normal images and their adversarial images generated by FGSM.

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
RBF-SVM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.8340	0.8913	0.9305	0.9487
adversarial images	0.8258	0.8936	0.9243	0.9541

Table 2: Detection accuracy of normal images and their adversarial images generated by IGSM.

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
RBF-SVM	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
normal images	0.7749	0.8660	0.9145	0.9362
adversarial images	0.7975	0.8752	0.9072	0.9330

Table 3: Detection accuracy of normal images and their adversarial images generated by Deepfool.

	normal images	adversarial images
SPAM	0.8553	0.8481
ESPAM	0.8870	0.8629
SRM	0.9445	0.9491
ESRM	0.9498	0.9527
RBF-SVM	0.5838	0.6012

Table 4: Detection accuracy of normal images and their adversarial images generated by C&W.

	normal images	adversarial images
SPAM	0.6957	0.6778
ESPAM	0.8025	0.8296
SRM	0.8814	0.9092
ESRM	0.9233	0.9341
RBF-SVM	0.5332	0.5187

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 IGSM, 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 set $\kappa = 0$. In the process of estimating MPM, we set $L = 100$.

At first, the 40000 images from ImageNet-1000 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.

Carlini & Wagner (Carlini and Wagner 2017a) point out that it is necessary to evaluate defenses using a strong attack on harder datasets (such as Imagenet). Moreover, Car-

lini & Wagner prove that using a second neural network to identify adversarial examples is the least effective defense. Therefore we only compare our method with the defense which is effective for C&W on Imagenet and not based on another neural network. However, Li's defense (Li and Li 2017) fails against the C&W attack. The Hendrycks defense (Hendrycks and Gimpel 2016) is only effective for MNIST. Grosse's (Grosse et al. 2017), Gong's (Gong, Wang, and Ku 2017) and Metzen's (Metzen et al. 2017) defenses use a second neural network to classify images as normal or adversarial. Lu's defense (Lu, Issaranon, and Forsyth 2017) has good performance on Imagenet even though its performance against C&W is not evaluated. Finally we compare our detectors with Lu's defense which is denoted as RBF-SVM.

The experimental results of detecting adversarial examples are shown in Table $1 - 4$. The data of Table $1 - 4$ is the detection accuracy of normal images and adversarial images. Figure 2 and Figure 3 illustrate these detectors' performance by averaging the accuracy of detecting normal images and adversarial images. First of all, the results reveal that the detectors with MPM have higher detection accuracy. Moreover, MPM has stronger enhancing effect on SPAM than SRM. When detecting FGSM and IGSM, 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 accuracy when detecting C&W. However, MPM improves SPAM by more than 15 percent and the detection accuracy of ESRM reaches 93 percent on detecting adversarial examples yielded by C&W. In addition, the detection accuracy of ESRM is the highest when detecting FGSM, IGSM, Deepfool and C&W. However, the computation time of SRM and ESRM is much longer because of their high-dimensional features.
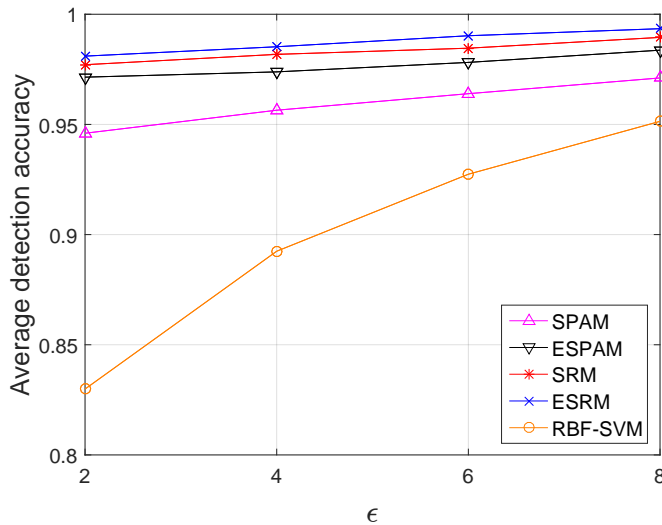

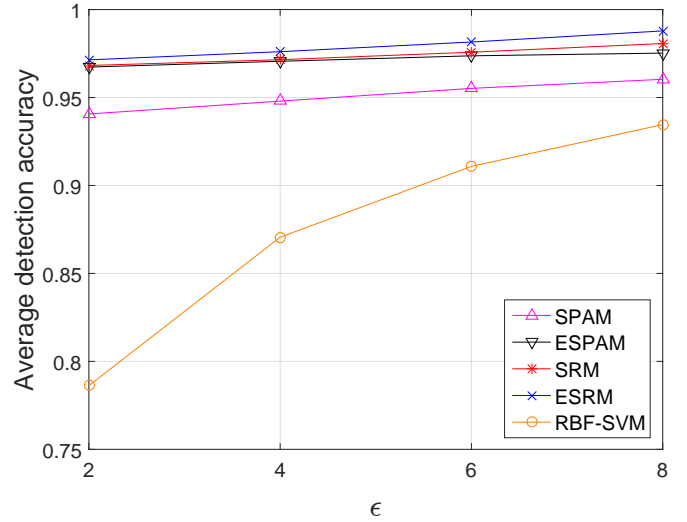Figure 3: Average detection accuracy for detectors against IGSM.

Conclusions

Inspired by the insight of Goodfellow et al. (Goodfellow, Shlens, and Szegedy 2015) 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 (Carlini and Wagner 2017a) cannot be directly performed to our method because the structure of our detection model is not a neural network. Therefore an open problem is how to implement secondary attacks on our proposed defense method.

References

[Amodei et al. 2016] Amodei, D.; Anubhai, R.; Battenberg, E.; Case, C.; Casper, J.; Catanzaro, B.; Chen, J.; Chrzanowski, M.; Coates, A.; Diamos, G.; et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, 173–182.

[Athalye, Carlini, and Wagner 2018] Athalye, A.; Carlini, N.; and Wagner, D. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*.

[Bhagoji, Cullina, and Mittal 2017] Bhagoji, A. N.; Cullina, D.; and Mittal, P. 2017. Dimensionality reduction as a defense against evasion attacks on machine learning classifiers. *arXiv preprint arXiv:1704.02654*.

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

[Carlini and Wagner 2017a] Carlini, N., and Wagner, D. 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th*

Figure 2: Average detection accuracy for detectors against FGSM.

ACM Workshop on Artificial Intelligence and Security, 3–14. ACM.

[Carlini and Wagner 2017b] Carlini, N., and Wagner, D. 2017b. Towards evaluating the robustness of neural networks. In *Security and Privacy*.

[Das et al. 2017] Das, N.; Shanbhogue, M.; Chen, S.-T.; Hohman, F.; Chen, L.; Kounavis, M. E.; and Chau, D. H. 2017. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*.

[Denemark et al. 2014] Denemark, T.; Sedighi, V.; Holub, V.; Cogranne, R.; and Fridrich, J. 2014. Selection-channel-aware rich model for steganalysis of digital images. In *Information Forensics and Security (WIFS), 2014 IEEE International Workshop on*, 48–53. IEEE.

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

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

[Goodfellow, Shlens, and Szegedy 2015] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICML*, 1–10.

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

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

[He et al. 2016] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

[Hendrycks and Gimpel 2016] Hendrycks, D., and Gimpel, K. 2016. Early methods for detecting adversarial images. *arXiv preprint arXiv:1608.00530*.

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

[Kurakin, Goodfellow, and Bengio 2017] Kurakin, A.; Goodfellow, I.; and Bengio, S. 2017. Adversarial examples in the physical world. *International Conference on Learning Representations*.

[Li and Li 2017] Li, X., and Li, F. 2017. Adversarial examples detection in deep networks with convolutional filter statistics. In *ICCV*, 5775–5783.

[Lu, Issaranon, and Forsyth 2017] Lu, J.; Issaranon, T.; and Forsyth, D. 2017. Safetynet: Detecting and rejecting adversarial examples robustly. In *IEEE International Conference on Computer Vision*, 446–454.

[Metzen et al. 2017] Metzen, J. H.; Genewein, T.; Fischer, V.; and Bischoff, B. 2017. On detecting adversarial perturbations. *International Conference on Learning Representations*.

[Moosavidezfooli, Fawzi, and Frossard 2015] Moosavidezfooli, S. M.; Fawzi, A.; and Frossard, P. 2015. Deepfool: A simple and accurate method to fool deep neural networks. In *Computer Vision and Pattern Recognition*, 2574–2582.

[Papernot et al. 2016a] Papernot, N.; Mcdaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016a. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy*, 372–387.

[Papernot et al. 2016b] Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016b. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, 582–597. IEEE.

[Papernot et al. 2017] Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 506–519. ACM.

[Pevny, Bas, and Fridrich 2010] Pevny, T.; Bas, P.; and Fridrich, J. 2010. Steganalysis by subtractive pixel adjacency matrix. *IEEE Transactions on information Forensics and Security* 5(2):215–224.

[Russakovsky et al. 2015] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115(3):211–252.

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

[Sullivan et al. 2005] Sullivan, K.; Madhow, U.; Chandrasekaran, S.; and Manjunath, B. S. 2005. Steganalysis of spread spectrum data hiding exploiting cover memory. In *Electronic Imaging 2005*, 38–46. International Society for Optics and Photonics.

[Szegedy et al. 2014] Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. *International Conference on Learning Representations*.

[Zheng et al. 2016] Zheng, S.; Song, Y.; Leung, T.; and Goodfellow, I. 2016. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4480–4488.