A Detect-and-Modify Region-based Classifier to Defend Evasion Attacks

Jiawei Jiang[1], Yongxin Zhao[*1,2], Xi Wu[3] and Genwang Gou[1]
[1] **Shanghai Key Laboratory of Trustworthy Computing,**
[2] **National Trusted Embedded Software Engineering Technology Research Center,**
East China Normal University, Shanghai 20062, China
[3] **The University of Sydney, Australia**

Abstract.
Deep Neural Networks (DNNs) are powerful models that have achieved impressive results on image classifications. However, they are vulnerable to be attacked by adversarial examples, which are crafted to cause prediction errors in DNNs. In order to make the networks more robust and reliable, in this paper, we present an improved region-based classification to mitigate the evasion attack, which is well known to attack DNNs via generating adversarial examples. Specifically, in our framework, an image is considered as a matrix of Markov Chains and we detect possible adversarial examples according to the Image Transition Probabilities (ITPs) in Markov Chains. Furthermore, we modify the original ITPs of the detected adversarial examples by using the saliency map of ITPs, and we employ our improved region-based classification on these updated adversarial examples to get a better output prediction. Finally, our experiments illustrate that our approach reduces the test errors imposed by adversarial examples on MNIST datasets and CIFAR-10 datasets.

1 Introduction

In recent years, deep learning has achieved remarkable results in image classification [5], malware detecting [3]. Though deep networks have exhibited a very good performance in classifications, researchers from the machine learning area have found a fatal weak point that Deep Neural Networks (DNNs) are extremely vulnerable to be attacked by adversarial examples [14]. In 2014, Szegedy et al. [14] firstly proposed the concept of adversarial example, which is a kind of images crafted by adding tiny perturbations into normal images (i.e., normal examples). For instance, an attacker may add a small noise into a test example such that it can deceive the state-of-the-art classifiers into giving an incorrect classification, which is also called the evasion attack. Recently, many algorithms, including FGSM [4], JSMA [12], have been proposed to generate adversarial examples, which make the rate of wrong predictions given by DNNs classifiers much higher. Moreover, because of transferability [11], an adversarial example generated by one DNN model is also able to mislead other DNN models which may have different network structures and physical attacks [6]. Thus, adversarial examples significantly limit the use of deep learning, especially in safety critical applications such as self-driving cars [9]. It is important for us to develop an approach to defend against adversarial examples and to mitigate the evasion attack from DNNs.

* Corresponding Author: yxzhao@sei.ecnu.edu.cn

DOI reference number: 10.18293/SEKE2020-039

To defend against evasion attacks, some defense methods have been proposed such as detecting adversarial example, adversarial training and distillation defense [7]. The method of detecting adversarial example is quite straightforward that the detection model determines whether an input is legal or not according to the differences between adversarial examples and normal examples. If the input is illegal, an exception handling strategy will be executed. However, it is usually difficult to design a proper exception handling strategy. The state-of-the-art detecting adversarial example method, which considers an image as a Markov Process, was proposed by Zhou et al [15]. Additionally, we can also mitigate evasion attacks by enhancing the robustness of networks themselves. Goodfellow et al. [4] used adversarial examples to train DNNs models, which is well known as adversarial training. Besides, Papernot et al. [13] proposed a distillation method to make DNNs robust against adversarial attacks, which uses the knowledge of the network to improve its own robustness. However, all these methods above sacrifice the classification accuracy of normal examples.

In 2018, Cao et al. [1] proposed the region-based classification algorithm, which samples some examples from a hypercube area centered by a test example and selects the label which appears most as the prediction result of the test example after predicting labels of all these sample examples via a trained DNNs model. Even though this algorithm not only maintains a high accuracy of DNNs classifiers on normal examples, but also increases the robustness of DNNs classifiers against adversarial examples, it in fact heavily relies on the surroundings of test examples. Specifically, the most sample examples from the surroundings of the test example can be classified by the DNNs model, the more reliable and robust of the prediction result of the algorithm on the existing evasion attacks. In the case shown in Figure 1, the region-based classification algorithm will behave badly because most sample examples from the surroundings (a hypercube area in blue) of the test example x' are outside the classification boundary (a curve in red) and they cannot be classified by the DNNs model. In this paper, we propose a new algorithm to improve the region-based classification algorithm for image classifications. An image is regarded as a matrix of Markov Processes, in which each Markov Process corresponds to one row of this image. Firstly, we define the Image Transition Probability (ITP) to represent the transformation probabilities of all image pixel values, and we perform 8000 MNIST data with adversarial examples and normal examples to illustrate that ITP can remarkably distinguish adversarial examples from normal examples. Secondly, we use ITP to detect

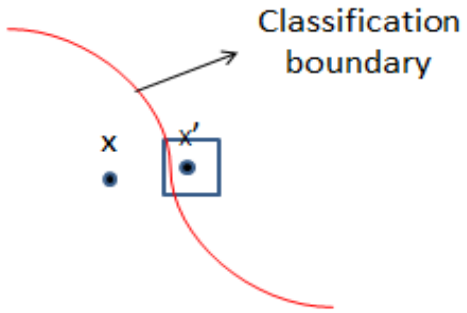

Figure 1. A case where the region-based classification behaves badly. Here, x is a normal example whereas x' is an adversarial example.

adversarial examples in an image, and decrease the original ITP of detected adversarial examples in order to make the examples locate at a proper location and to make the process more reliable. Finally, we observe that it is easy to sample some examples with much noise when sample examples from hypercube centered by the test example. Thus, we newly create an area centered by the test example which is different from the one used in the region-based classification.

In summary, the main contributions of our work include:

- We propose a new algorithm to address dependency on the surrounding of the test example through modifying the detected possible adversarial examples. As far as we know, it is the first work to combine the region-based classification with detecting method.
- We give a new region centered by the test example to sample examples which are different from region-based classification. Our region is anisotropic and the region given by region-based classification is isotropic.

The structure of the remaining paper is given as follows. Section 2 shows some preliminaries of this paper including a brief introduction on evasion attacks and some defense methods. Section 3 describes our improved region-based classification approach. Our experiments are in Section 4, which demonstrate the effectiveness of our algorithm. In Section 5 we conclude our work and outlook future works.

2 Preliminaries

We firstly introduce the following notations:

- Let X be the normal example without perturbation.
- Let X' represent the adversarial example which is designed by attacker to make classifiers wrong.
- Let $\nabla J_X(X, \theta, y)$ denote the gradient of cost function in DNNs. Where, X is the input, y is the true class and θ is the parameter of neural network.
- We use $sign(m)$ to represent sign function which returns the sign of a certain number m (positive or negative). If m is metric, it will return a matrix which consists of -1, 1 and 0.
- Let ϵ represents a small numerical.

Now, we will give a brief introduction on several typical attacks and some major defense methods which have been used in our experiments.

Evasion Attacks. Evasion attacks are well-known attacks in machine learning area which generate adversarial examples to attack machine learning models. There are two types of evasion attacks, including target evasion attacks and no-target evasion attacks. In target attack, an adversarial example is designed to make classifiers to give a wrong prediction. In contrast, an adversarial example is only generated to make classifiers wrong in the no-target attacks. In this paper, we only discuss no-target attacks.

1) **Fast Gradient Sign Method (FGSM)**: It is one of the simplest methods to get adversarial examples by calculating the gradient of cost function with respect to the input X, which is motivated by the linear nature of DNNs models. Goodfellow et al. [4] efficiently get the adversarial example X' via the following equation:

$$X' = X + \epsilon \cdot sign(\nabla J_X(X, \theta, y))$$

2) **Jacobian-based Saliency Map Attack (JSMA)**: JSMA is a method to create an adversarial example by restricting the l_0-norm of the perturbations, proposed by Papernot et al. [12]. It computes a saliency map by extracting some important pixels from the input image in which a small change can make the output various, and then modifies these pixels iteratively until getting the output variously.
3) **Carlini Wagner (CW)**: CW [2] is an efficient algorithm which generates successful adversarial examples with small noise. It shows that an adversarial example generated by the CW method is also effective in defensive distillation networks. Considering three constraints l_0,l_2 and l_∞, CW attacks can mainly divided into three attacks as well, i.e., l_0 attack, l_2 attack and l_∞ attack. In our paper, we mainly use l_∞ attack in the experiments.
4) **DeepFool**: DeepFool is a no-targeted attack proposed by Moosavi-Dezfooli et al [10]. The key idea of DeepFool is to generate adversarial examples iteratively, that is each step searches a decision boundary direction to modify examples. It can generate adversarial examples with minimum noise.

Defense Methods Against Evasion Attacks. Adversarial examples have brought great threats to security applications in deep learning area. In order to defend against adversarial examples, various defense methods have been advised.

1) **Adversarial training**: Firstly introduced by Goodfellow et al., it is an effective defense method which trains the model via augmenting the training datasets with adversarial examples. Specifically, adversarial training can be considered as model regularization by adding the loss function of adversarial examples to DNNs models. Since adversarial training achieves a great effect in dealing with adversarial examples, several variants of it have been proposed with respect to different adversarial attack algorithms. Although adversarial training has the state-of-the-art defending performance, it still has a limitation. Adversarial training is found that it decreases the classification accuracy on normal examples. For instance, the DNNs model without adversarial training performs greater than that with adversarial training on CIFAR-10.

2) **Region-based classification**: Although many defense methods have been proposed, they have the same issue that they sacrifice the accuracy on normal examples. The region-based classification achieves a high accuracy on normal examples and also enhances the robustness of DNNs models. The key idea in region-based classification is sampling some examples which can help DNNs models predict the results on test examples. When testing an example, it samples some examples from a small region centered at this example and uses a trained DNNs classifier to test these examples. The suggested class of this example is the most voted by the classifier. However, region-based classification cannot behave well on FGSM attacks, which only gets about 10 percent accuracy. It is considered that this method is easy to be attacked by high-distortion adversarial examples. In this paper, we propose an improved region-based method to solve this problem.

3 Detect-and-modify region-based classifiers

The region-based classification proposed by Cao et al. [1] depends on the region centered by the test example. If a test example locates at the area where most of examples cannot be classified correctly by the DNNs model, the region-based classification then behaves badly. In this section we mainly present our detect-and-modify region-based classifiers to mitigate evasion attacks. There are three main processes

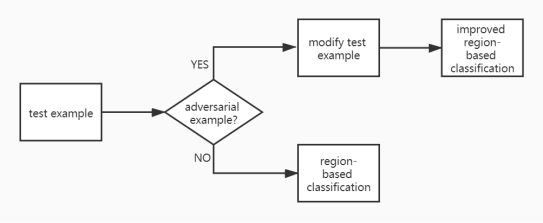

Figure 2. Flow of our approach

in our method, including detecting adversarial examples, modifying adversarial examples and an improved region-based classification. Figure 2. shows the whole process of our approach.

When testing an example, we firstly employ ITP, which can significantly distinguish adversarial examples from normal examples, to determine whether this test example is an adversarial example or not. Specially, if the ITP of this example is greater than the threshold value derived from training datasets, this example is supposed to be an adversarial example.

Secondly, if this example is an adversarial example, we use the saliency map of ITP to modify its original ITP. Thus the updated example facilitates to decrease its ITP and enhance the reliability.

Thirdly, we use our improved region-based classification to predict the label of the test example. However, if the test example is a normal example, we simply use the traditional region-based classification to predict its label.

The pseudo code of our algorithm is listed in Algorithm 1. There are four main procedures used by our approach. Procedure **detect**(x) determines whether x is an adversarial example or not. Procedure **modify**(x) carries out the modification of x if x is detected as an adversarial example and returns x'. Procedure **IRC**(x') proposed by us, gives the label of x. Procedure **RC**(x) predicts the label for x.

Algorithm 1 Detect-and-modify region-based classification

Input: A test example x
Output: The prediction label y_x of x

```
Flag = detect(x)
if Flag == true then
    x' = modify(x)
    y_x = IRC(x')
else
    y_x = RC(x)
end if
return y_x
```

3.1 Detecting adversarial examples

Firstly, we introduce our detecting process. In an image, there are many pixels and a pixel is usually related to its adjacent pixel in the same row. In this paper, we consider a single relation that the next pixel is related to its former pixel. For an image, we consider one image row as a Markov Chain and the pixel in the same row is equal to the discrete state of the Markov Chain. Therefore, the image can be considered as a structure with many Markov Chains. The space of state in a Markov Chain, which is consist of the value of pixels, ranges from 0 to 255. The ITP which we firstly proposed represents the transformation probability of all image pixel values related to adjacent pixels in the same row. The ITP value can be calculated by the following formula:

$$ITP = \sum_{h=1}^{H} P_h(x_2|x_1) \times P_h(x_3|x_2) \times \cdots \times P_h(x_w|x_{w-1}),$$

where $P_h(x_{i+1}|x_i)$ is the probability of adjacent pixels in h-th image row and w represents the number of pixels in one image row. $P_h(x_{i+1}|x_i)$ can be gotten in state transition which is calculated by the normal examples. Then, we calculate the ITP by adding all Markov chain transitions in an image. We put a single-channel image as an example.

Suppose a single-channel image has H rows and W columns, which means it includes H Markov Chains and W discrete states in each chain. Each row of image can be converted into one dimensional vector X with W pixels. So an image is composed by a set of X_1,X_2,...,X_m. For a X_m, x_m^n represents n-th pixel in X_m. Let i represent the value of x_m^n range from 0 to 255. The x_m^{n+1} is a pixel which is next to x_m^n and its value denotes j. While $P(i,j)$ represents the transition probability from state value i to state value j. In other words, $P_{i,j}$ is the probability of appearance of value pair(i,j) at two adjacent pixels. So, the transition of probability matrix P can be described as follows:

$$P = \begin{bmatrix} P_{0,0} & \cdots & P_{0,255} \\ \vdots & \ddots & \vdots \\ P_{255,0} & \cdots & P_{255,255} \end{bmatrix}$$

We can calculate the transition of probability matrix P based on training sets as below:

$$P_{i,j} = \frac{\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{t=1}^{W-1} h_{i,j}(x_t, x_{t+1})}{\sum_{n=1}^{N}\sum_{j=0}^{255}\sum_{h=1}^{H}\sum_{t=1}^{W-1} h_{i,j}(x_t, x_{t+1})}$$

Here, $h_{i,j}(x_t, x_{t+1})$ will be 1 only if the value of two adjacent pixels is (i,j). Otherwise, it will be 0. H denotes the number of image rows and N is the number of examples in the training set. Then,

we can calculate one image row which is considered as a markov chain transition probability ($IRTP$) value by the following formula:

$$IRTP_h = P_h(x_1, x_2) \times P_h(x_2, x_3) \times \cdots \times P_h(x_{W-1}, x_W)$$

where $P_h(x_i, x_{i+1})$ is the probability of adjacent pixels which the first pixel is i-th position pixel in h-th row. However, the value of P is so small that we use log in $IRTP$.

$$IRTP_h = \log P_h(x_1, x_2) + \log P_h(x_2, x_3) + \cdots + \log P_h(x_{W-1}, x_W)$$

The ITP is the sum of all $IRTP$ value.

$$ITP = \sum_{h=1}^{H} IRTP_h$$

Because of adding small perturbation, an adversarial example breaks Markov process nature for image row pixels. Thus, the ITP of adversarial examples is bigger than the one of normal examples. Through experiment results showed in Figure 3, we can find that the ITP efficiently distinguishes between adversarial examples and normal examples.

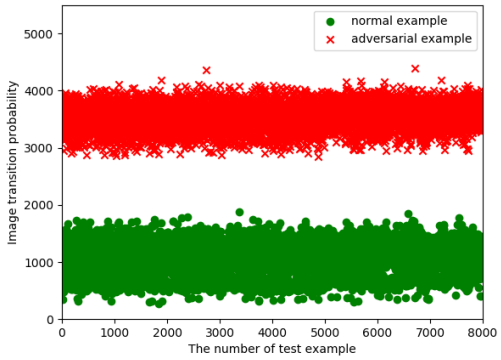

Figure 3. ITP on mnist datasets, red color represents adversarial examples and green color represents normal examples

Thus, the ITP is a sum of all markov chain transition probabilities which we consider a image row as a markov chain. Of course, image transition probability also can be got by considering one image column as a markov chain. So, ITP_r represents image transition probability which consider image row as markov chain. ITP_c represents image transition probability which consider image column as markov chain. In our work, image transition probability we have proposed denotes ITP_r. During detecting period, we compute the image transition probability of test example. If ITP of this example more than a the threshold ITP_m which can be got from training dataset, this example maybe adversarial example and need to be modify. Especially, we transform it into one channel image if this image is multi-channel image. The process of detecting is showed in algorithm 2.

3.2 Modifying adversarial examples

Next, we give the process of modifying image after detecting test example which is a possible adversarial example. It is difficult to sample some examples which can be correctly predicted for a adversarial

Algorithm 2 Procedure **Detect**

Input: A test example x and a threshold value ITP_m
Output: Boolean value

```
compute the image transition probability of x as ITP_x
if ITP_m ≤ ITP_x then
    return true
else
    return false
end if
```

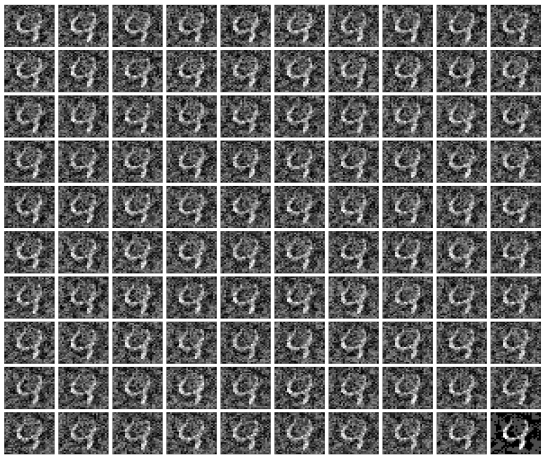

Figure 4. The figure shows the samplers from small region centered at adversarial examples, At last in the figure is adversarial examples which generated by FGSM

example. As figure 4 showed, the examples sampled around adversarial example is extremely unclear and obscure which has a large number of noise and can not recognized by us. So it is necessary to modify image to make the sampled examples with small noise. And we found that the image transition probability of adversarial example is bigger than normal example's. The image transition probability describes the difference on image noise. If the image has much noise, The image transition probability value will be big. Thus, we can modify image by making its image transition probability smaller before region-based classification.

The key idea of modify image is how to choose suitable pixels to change and how to select suitable values to represent for chose pixels so that make the image transition probability decrease. To address this problem, we firstly use saliency map to describe the impact of each pixel in an image. The saliency map can be got by following equation

$$S_{r,m,i,n} = ITP_{r,n} - ITP_{r,o}$$
$$S_{c,m,i,n} = ITP_{c,n} - ITP_{c,o}$$

where m represents m-th row or column. r denotes row. i is i-th pixel in one markov chain. The n denotes pixel value. So $ITP_{r,n}$ is the value of image transition probability which consider image row as a markov chain if current pixel value is n. ITP_o represents the value of image transition probability without modifying. $S_{r,m,i,n}$ denotes the change of image transition when i-th pixel value turn into n in m-th row. The $S_{c,m,i,n}$ represents the change of ITP_c that consider image column as a markov chain. Then, we choose pixels to modify with min $S_{r,m,i,n}$ in order to decrease ITP_r quickly. However the ITP_r of edge pixels which we do not need to modify are obviously bigger than the others. In other word, only when the all

$s_{r,m,i,n}$ is not exceed threshold ITP_t, we should choose suitable value to represent original value. this threshold ITP_t can be got in training datasets by search min value of all edge pixels. In our experiments, we take $ITP_t - 2$ for MNIST datasets. Another challenge is choose suitable pixel n for chose pixel. we search n from 0 to 255 and choose n which satisfy min $S_{r,m,i,n}$ and $S_{c,m,i,n}$ is not more than 0.

3.3 Improving region-based classification

At last, we give a method called improved region-based classification after modifying detected adversarial examples. Region-based classification randomly uniformly sample some examples from the hypercube of test examples x and help test example predict its label. this hypercube can formally defined: $B(x,l) = \{y|y_i \in [0,1], |y_i - x_i| \leq l, \forall i = 1,2,...,n\}$, where y_i and x_i are the i-th dimensions of x and y. However, as figure 4 showed, the examples sampled centered at adversarial example is extremely unclear and can not recognized by us.

So we need to create a new area which can promise the example with small noise when test example is detected as adversarial example. we denote the new area $A(x,r)$ which is centered at test example x which we consider it as a matrix and different $x_{i,j}$ has different length $r_{i,j}$. Formally, $A(x,r) = \{y|y_{i,j} \in [0,1], |y_{i,j} - x_{i,j}| \leq r_{i,j}, \forall j = 1,2,...,n\}$, where i are the i-th row of metric x,metric y, metric r and j are the j-th column of metric x, metric y, metric r. Choosing area centered at test example is equal to determine the metric r on test example x. Specifically, we learn the length l in r for every dimension of metric x though a search process described in algorithm 4. The key idea in search process is selected the max length l such that the image transition probability of example sampled from $A(x,r)$ is all smaller than the test example's. In our work, it will be modified to be x' if it is detected as adversarial example for a test example x. Initially, we set $r_{i,j}$ which is corresponding to $x_{i,j}$ be a small value. Then we increase $r_{i,j}$ until image transition probability is bigger than x.

Algorithm 3 Learning metric r by Searching

Input: A text example x with H rows and W columns, ITP value ITP_x of x and an updated image x' after modifying the ITP

$ITP_{x',i,j,z}$ denotes the image transition probability of x' when $x'_{i,j}$ becomes z.
$l = r_0, i = 0, j = 0$
while $i \leq H$ **do**
  while $j \leq W$ **do**
    $z1 = x'_{i,j}$, $z2 = x'_{i,j}$
    while $ITP_{x',i,j,z1} \leq ITP_x$ and $ITP_{x',i,j,z2} \leq ITP_x$ **do**
      $r_{i,j} = r_{i,j} + 1$
      $z1 = x'_{i,j} + r_{i,j}$, $z2 = x'_{i,j} - ri, j$
    end while
    $j = j + 1$
  end while
  $i = i + 1$
end while

4 Experiments

This section introduces our experiments on two common image datasets with our algorithm and compared methods.

4.1 Experiment setup

Datasets: we test our algorithm on two standard image datasets : MNIST and CIFAR-10. MNIST datasets includes a handwritten numbers images with 50000 training examples and 10000 testing examples which size is 28*28. CIFAR-10 has a total of 60,000 color images. These images are 32*32, divided into 10 categories, each category has 6000 images.

Nerual Networks: In our experiments, we use a common convolutional neural network described as following:

Layer type
Convolution + Relu
Max pooling
Convolution + Relu
Max Pooling
Convolution + Relu
Max Pooling
Fully connected
Softmax

compared methods: We use four algorithms to generate adversarial example, such as JSMA, DeepFool, FGSM and CW. Especially, we take ϵ 0.2 in FGSM algorithm. And we use l_∞ to generate adversarial examples in CW algorithm. Then we Compare our method with the following classifiers in same adversarial examples.

- **adversarial training**:First, we use adversarial training to learn a DNN model for each datasets. Recently, A popular adversarial training algorithm which was proposed by Madry et al [8] use PGD algorithm to generate adversarial examples and learn the DNN classifier with these adversarial examples and normal examples. However, These adversarial examples have so much noise that sacrifices classification accuracy significantly. Thus, we adopt DeepFool to get adversarial examples in adversarial training.
- **region-based classification**: For each datasets, we train a DNN classifier which structure is the same as adversarial training's. According to Cao et al work. we respectively set r which determine the size of region centered at test example to be 0.3 in MNIST datasets and take r 0.02 for CIFAR-10 datasets. In addition, we sample 100 examples in region-based classification for each testing example.
- **our methods**: The architecture of nerual networks is same with compared method's. we set ipt_m is 1800 in MNIST datasets and 2000 for CIFAR-10 during detecting adversarial examples. During region-based classification, we sample 100 examples for each testing example.

4.2 Results

First, we perform a experiment which test normal example on our methods and compared methods which test 5000 examples from MNIST testing datasets and CIFAR-10 datasets. The result is showed in Table 1 which show classification accuracy on normal examples. From the table 1, we can find that our methods get better accuracy on normal examples than adversarial training and achieve the same accuracy with standard networks. Next, we test totally 4000 adversarial examples of MNIST datatsets generated on testing datasets for region-based classification and our methods. These adversarial examples is generated by FGSM, JSMA, CW and DeepFool which each methods generated 1000 adversarial examples. The result is showed

Table 1. Classification accuracy on normal examples

	MNIST	CIFAR-10
standard CNNs	99.1%	89.8%
Adversarial training	98.3%	87%
region-based classification	99.1 %	89.8%
our methods	99.1 %	89.8%

in table 2. It showed that our methods perform better than region-based classification on mnist datasets. Especially in CW attack, our methods can evasion this attack.

Table 2. Classification accuracy on adversarial example

Method	MNIST datasets		CIFAR-10 datasets	
	region-based classification	our method	region-based classification	our method
FGSM	16.2%	54.5%	13.4%	52.0%
DeepFool	5.1 %	59.8%	10.6 %	50.3%
JSMA	7.3 %	46.2%	6.0 %	56.7 %
CW-L_∞	23.1 %	79.8%	16.5 %	73.7%

In the end, we compare region-based classification and our method on CIFAR-10 datasets. CIFAR-10 datasets is a three channel image set. First, we use CIFAR-10 trianing datasets train a CNNs and randomly sample 1000 examples from CIFAR-10 testing datasets as testing examples. Then we use FGSM , CW and DeepFool methods to generate adversarial examples based on testing examples. Especially, we transform three channel image into one channel when carry out our detecting methods. Though experiments, as table 2 showed, our methods also achieve good accuracy on adversarial examples.

5 Conclusion and Future work

In this work, we propose a detect-and-modify region-based classifier to mitigate evasion attacks in DNNs. Firstly, we observe that region-based classification is limited by the surroundings of examples (especially the adversarial examples). Thus we use ITP, which can remarkably distinguish adversarial examples from normal examples, to detect whether the test example is an adversarial example or not. In order to improve the accuracy on adversarial examples, we modify the ITP of the test example which may be an adversarial example by decreasing its original ITP. Then we use our improved region-based classification to predict results on test examples.

We apply our approach on experiments with different adversarial examples generated by different methods and different datasets. The experimental results show that our method behave better than the traditional region-based classification. As far as we know, our work is the first work to improve the region-based classification by combining it with detecting methods. The improved method finally solves the issue that the region-based classification is limited by the surroundings of test examples. In the future, we will continue improving our method to make it better to modify images with a large perturbation.

6 Acknowledgement

This paper is partially supported by National Key Research and Development Program of China (Grant Nos. 2019YFA0706400), Science and Technology Commission of Shanghai Municipality Project (No. 18ZR1411600) and the Open Project of Shanghai Key Laboratory of Trustworthy Computing (No. 08dz22304201804).

REFERENCES

[1] Xiaoyu Cao and Neil Zhenqiang Gong, 'Mitigating evasion attacks to deep neural networks via region-based classification', in *Proceedings of the 33rd Annual Computer Security Applications Conference, Orlando, FL, USA, December 4-8, 2017*, pp. 278–287, (2017).

[2] Nicholas Carlini and David A. Wagner, 'Towards evaluating the robustness of neural networks', in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 39–57, (2017).

[3] Bingcai Chen, Zhongru Ren, Chao Yu, Iftikhar Hussain, and Jintao Liu, 'Adversarial examples for cnn-based malware detectors', *IEEE Access*, **7**, 54360–54371, (2019).

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, 'Explaining and harnessing adversarial examples', in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, (2015).

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, 'Imagenet classification with deep convolutional neural networks', in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pp. 1106–1114, (2012).

[6] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, 'Adversarial examples in the physical world', in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, (2017).

[7] Qiang Liu, Pan Li, Wentao Zhao, Wei Cai, Shui Yu, and Victor C. M. Leung, 'A survey on security threats and defensive techniques of machine learning: A data driven view', *IEEE Access*, **6**, 12103–12117, (2018).

[8] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, 'Towards deep learning models resistant to adversarial attacks', in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, (2018).

[9] Roland Meier, Thomas Holterbach, Stephan Keck, Matthias Stähli, Vincent Lenders, Ankit Singla, and Laurent Vanbever, '(self) driving under the influence: Intoxicating adversarial network inputs', in *Proceedings of the 18th ACM Workshop on Hot Topics in Networks, HotNets 2019, Princeton, NJ, USA, November 13-15, 2019*, pp. 34–42, (2019).

[10] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, 'Deepfool: A simple and accurate method to fool deep neural networks', in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2574–2582, (2016).

[11] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow, 'Transferability in machine learning: from phenomena to black-box attacks using adversarial samples', *CoRR*, **abs/1605.07277**, (2016).

[12] Nicolas Papernot, Patrick D. 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, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pp. 372–387, (2016).

[13] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, 'Distillation as a defense to adversarial perturbations against deep neural networks', in *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pp. 582–597, (2016).

[14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus, 'Intriguing properties of neural networks', in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, (2014).

[15] Yue Zhou, Xiaofang Hu, Lidan Wang, Shukai Duan, and Yiran Chen, 'Markov chain based efficient defense against adversarial examples in computer vision', *IEEE Access*, **7**, 5695–5706, (2019).