IET Computer Vision

Research Article

IET Journals
The Institution of Engineering and Technology

Adversarial examples detection through the sensitivity in space mappings

ISSN 1751-9632
Received on 21st May 2019
Revised 20th January 2020
Accepted on 24th February 2020
doi: 10.1049/iet-cvi.2019.0378
www.ietdl.org

Xurong Li[1] ✉, Shouling Ji[1], Juntao Ji[1], Zhenyu Ren[1], Chunming Wu[1], Bo Li[2], Ting Wang[3]

[1]*Department of Computer Science and Technology, Zhejiang University, Hangzhou, People's Republic of China*
[2]*Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, USA*
[3]*Department of Computer Science, Lehigh University, Bethlehem, USA*
✉ *E-mail: lixurong@zju.edu.cn*

Abstract: Adversarial examples (AEs) against deep neural networks (DNNs) raise wide concerns about the robustness of DNNs. Existing detection mechanisms are often limited to a given attack algorithm. Therefore, it is highly desirable to develop a robust detection approach that remains effective for a large group of attack algorithms. In addition, most of the existing defences only perform well for small images (e.g. MNIST and Canadian institute for advanced research (CIFAR)) rather than large images (e.g. ImageNet). In this paper, the authors propose a robust and effective defence method for analysing the sensitivity of various AEs, especially in a much harder case (large images). Their method first creates a feature map from the input space to the new feature space, by utilising 19 different feature mapping methods. Then, a detector is learned with the machine-learning algorithm to recognise the unique distribution of AEs. Their extensive evaluations on their proposed detector show that their detector can achieve: (i) low false-positive rate (<1%), (ii) high true-positive rate (higher than 98%), (iii) low overhead (<0.1 s per input), and (iv) good robustness (work well across different learning models, attack algorithms, and parameters), which demonstrate the efficacy of the proposed detector in practise.

1 Introduction

In recent years, deep neural networks (DNNs) have been widely used in many crucial applications, such as auto-driving [1] and traffic sign classification [2]. Although DNNs have exhibited impressive performance on these tasks, it has been shown that they are vulnerable to well-crafted adversarial examples (AEs) [3–7], which are the inputs added with a small imperceptible perturbation and are typically generated by using an optimisation procedure. The existence of AEs hinders the application of DNNs. For example, an attacker may use a fake adversarial image to fool the face detection system, causing forgery of legitimate users, i.e. AEs based attack raises security concerns about the application of DNNs.

Current defences against AEs mainly follow two approaches: (i) one gives a complete classification, where it tries to make DNNs classify AEs correctly [8–15] and (ii) other conducts detection of AEs without modifying the DNNs [16–20]. The first approach can find known AEs well, but it usually needs to retrain the DNNs or modify the architecture of the networks. For instance, *adversarial training* [5, 21] and *defensive distillation* [9], which will need lots of AEs from various attacks and introduce huge training cost. In addition, the modified new DNNs usually remain vulnerable to *second AEs*, namely the attacker can always find the adversarial inputs through probing the decision boundary of the new network. The second approach is independent of the original DNNs. Before giving a prediction, the detection system will find suspicious inputs and block out of them to avoid the misclassification of DNNs. Since we target crucial security applications, such as face detection and malware detection, we focus on the second method, where detection before DNNs can resist the adversarial attacks from malicious attackers.

Recent literature on detection-based defences can be further divided into two categories. The first class is based on the secondary classification, which leverages the difference between AEs and normal examples to learn a classifier [16, 18]. The other line of work is exploring inconsistencies between AEs and normal examples in certain scenarios, such as statistical test [20], principal component analysis (PCA) weights [15], and squeezing skills [17]. However, there are several major limitations to these existing works. First, most existing methods were not comprehensively evaluated. They were only evaluated on small images such as modified national institute of standards and technology database (MNIST) and CIFAR10 [16, 18, 19]. Since attacking large images is easier than small images, on the other hand, the defences against attacks on large images (e.g. ImageNet) become harder. This is also confirmed by our experiments in Section 5. Second, most of defences are lacking robustness [17, 18, 20]. A good detection-based defence should work well across different attack algorithms or target models. In other words, no matter how the attack parameters, algorithms, and target model change, a robust defence should be adaptive to them. Unfortunately, previous detection-based defences are sensitive to the parameters and algorithms of attacks, and they are model specific. For example, feature squeezing [17] needs different thresholds for various AEs, and in [18], one classifier trained on specific AEs cannot detect AEs from other generation algorithms. Third, most of the existing detection methods cannot defend well against white-box attacks, where the adversary knows the detector and the target model. In [22–24], it has been proven that all the above-discussed detection methods can be bypassed easily. Hence, it is highly desirable to develop a robust and effective detection method, which can work well across different attack algorithms and target models. Even under white-box attacks, the original DNNs with the proposed detection strategy can still drastically reduce the attack success rate (ASR).

To address all of the issues above, in this work, we design an effective detector based on the sensitivity of AEs, which works well for more importantly large images. We first construct a new feature space by utilising 19 space mapping methods. Then, by extracting the labels and probability distributions of the corresponding DNN predictions, we train a detector in the new feature space. Our detector can learn the differences in the distribution between AEs and normal examples in the new space. Finally, we evaluate our detector against the state-of-the-art attacks.

It should be noted that our detector is significantly different from existing ensemble-based methods [17, 25]. Unlike the existing works focusing on the decisions based on multiple detectors, our method is to explore the sensitivity of AEs under various mapping methods. Furthermore, the space mapping

methods used in our detector are not limited to the squeezing skills in [17]. In contrast, we consider both labels and probability distributions of the prediction results. It has been shown experimentally that our detector is indeed achieving better detection on various AEs than feature squeezing [17].

In summary, this paper makes the following contributions:

- We systematically show that the evaluation of the existing detection method against AEs is far from sufficient. Most of them, which were evaluated on small images and performed well previously, are shown to perform poorly on large images. In addition, they cannot work well across different models and attack algorithms.
- We propose a novel sensitivity-based detector to overcome the weaknesses of existing detection approaches. By utilising 19 space mapping methods, the distribution of AEs can be distinguished from that of normal examples in the new feature space. Our detector makes a solid attempt to build a robust detector across different learning models and attack algorithms.
- We conduct a comprehensive evaluation of the proposed detector. The results show that the detector can achieve a high true-positive rate (TPR) (98%), whereas a low false-positive rate (FPR) (<1%), which significantly outperforms state-of-the-art detection approaches. In addition, the detector is very efficient (<0.1 s for detecting per input) and robust to adaptive attacks. Our detector can also be used to defend against white-box attacks. In particular, with our detector deployed, the success rates of white-box attacks drop below 5%.

2 Background and related works

2.1 Adversarial attacks on neural networks

Given a network f, and a natural input x so that $C[f(x)] = l$. When we write $C[f(x)]$, we mean the classification of f on x. There are two types of adversarial attacks. One is untargeted attack: we say that x' is an untargeted AE if $C[f(x')] \neq l$ and $\Delta(x, x') \leq \epsilon$, where $\Delta(x, x')$ quantifies the difference between x and x' and ϵ is usually a very small value. The other is targeted attack, which is usually harder than the untargeted attack: given a target class $t(t \neq l)$, we say that x' is a targeted AE if $C[f(x')] = t$ and $\Delta(x, x') \leq \epsilon$. To measure the difference between x and x', there are widely used distance metrics in previous literatures. Among them, the most popular one is as below:

$$\| v \|_p = \left(\sum_{(i=1)}^{n} \left| v_i \right|^p \right)^{1/p} \tag{1}$$

where $\| v \|_p$ means the L_p norm and $L_p = \| x - x' \|_p$. Usually, L_0, L_2 and L_∞ norms are used to measure the quality of AEs. In brief, the L_0 norm can measure the number of modified pixels between x and x'; L_2 norm can measure the Euclidean distance between x and x'; and L_∞ norm can measure the maximum change among the modified pixels between x and x'.

Next, we give a brief overview of several well known attacks, which form the basis for our experiments. In the following section, for simplicity, let AE and CE denote the adversarial and clean (normal) examples, respectively.

Fast gradient sign method (FGSM): FGSM is a single-step attack proposed in [5]. It is simple and uses the sign of the gradient of the loss function to compute the perturbation. Let θ be the parameters of the target model, x the input to the model, y the true label of x and $J(\theta, x, y)$ be the cost function used in training f. Then, the AE of FGSM is defined as

$$x^{\text{adv}} = x + \epsilon \, \text{sign}(\nabla_x J(\theta, x, y)) \tag{2}$$

where ϵ is a sufficiently small hyperparameter to be undetected. Note that FGSM is designed to be fast instead of generating close AE. On the basis of the FGSM attacks, momentum techniques can also be generalised to a targeted class, which is called targeted momentum iterative (MI)-FGSM [26].

Iterative gradient sign method (IGSM): Kurakin *et al.* [27] proposed an iterative version of FGSM, namely IGSM. Instead of taking a single step of size ϵ, multiple small sizes α are chosen, and the value is clipped to ensure that it is in an ϵ-neighbourhood of the original image

$$x_0^{\text{adv}} = x, \; x_i^{\text{adv}} = \text{clip}_{x,\,\epsilon}\left(x_{i-1}^{\text{adv}} + \alpha \, \text{sign}\left(\left(\nabla_{x_{i-1}^{\text{adv}}} J\left(\theta, \, x_{i-1}^{\text{adv}}, \, y\right)\right)\right)\right) \tag{3}$$

In our experiments, we set $\alpha = 1$ and the number of iterations to be $\min(\epsilon + 4, \, 1.25\epsilon)$ as shown in [27]. Besides, we can also set target classes during the iterations to launch targeted attacks, which are also called target gradient sign method (TGSM) attacks.

DeepFool: Moosavi-Dezfooli *et al.* [6] proposed DeepFool, an untargeted iterative attack, which approximates the classifier as a linear decision boundary and finds very small perturbations to cross the boundary. The perturbations caused by DeepFool are smaller than FGSM, and they are still effective to deceive the target models.

Carlini and Wagner (CW) Attacks: Carlini and Wagner [7] introduced new gradient-based attack algorithms (CW attacks), which can break defensive distillation [9]. Actually, CW attacks are similar to a range of attacks [7] and they all share the same optimisation framework. According to the distance metric, CW attacks can be categorised as L_0, L_1 and L_∞ attacks. Note that we use the L_2 norm in the evaluation, and we adopt a CW L_2 attack in this paper. Specifically, let t be the target class. Then search a perturbation δ that solves

$$\begin{aligned} &\text{minimise} \quad \| \delta \|_2^2 + c \cdot f(x + \delta) \\ &\text{such that} \quad x + \delta \in [0, \, 1]^n. \end{aligned} \tag{4}$$

Here, $f(\cdot)$ is defined as below:

$$f(x') = \max\left(\max\left\{ Z(x')_i : i \neq t \right\} - Z(x')_t, \; -\kappa \right) \tag{5}$$

where κ is a hyperparameter that controls the confidence of AE. A larger κ will make the attacker generate the AE, which is classified as t by the model with high confidence. Here, c is a hyperparameter, which can make a trade-off between $\| \delta \|_2^2$ and $f(\cdot)$. Besides, c can be tuned with binary search in the CW attack.

We do not consider the other attacks such as projected gradient descent attacks [21] and Jacobian-based saliency map attacks [4] since they do not scale well on large images, although they have been proven effective on small datasets such as MNIST and CIFAR10.

2.2 Defence against adversarial attacks

Many defensive techniques against adversarial attacks have been proposed recently. Although the previous literatures [28, 29] have provided different taxonomies on defences, we divide defences into two categories based on their results from this paper's perspective. The first one is complete classification, where the defences try to classify an AE correctly. The other is detecting AE, where it does not modify the model while detecting the abnormal AE only.

2.2.1 Complete classification: In the beginning, people attempt to classify AE correctly by modifying the original network architecture:

Adversarial training: In *adversarial training* [5, 21], the AEs, which are already known, are collected and added to the training data of DNNs. Then, the DNNs are retrained with the new training set to improve their robustness. However, such training may lead to a huge cost and a little drop in classification accuracy. Besides, adversarial training can only defend against the known attacks, and it cannot block the second attack, namely the attacker can still generate AE on the new model.

Input transformation: To avoid modifying the network architecture, several studies, which transform the input before the DNNs have

been proposed in recent years. Prakash *et al.* [11] introduced *pixel deflection*, which can locally corrupt the image by redistributing pixel values to make the AE become normal after this process. However, too complex transformation models are used in deflection and they only analyse the AE under black-box attacks. Xie *et al.* [12] pointed that *random resizing* and *random padding* can mitigate the adversarial effects, but it is costly to combine an adversarial training model with random layers. Afterwards, *bit-depth reduction*, *Joint Photographic Experts Group (JPEG) compression*, *total variance minimisation,* and *image quilting* are proposed to mitigate the adversarial effects of AE by researchers [13, 30]. Unfortunately, most of these transformation-based defences cannot take the AE with high confidence for evaluation and they suffer from accuracy loss with CE. More importantly, one transformation alone is insufficient to reverse the adversarial effect.

Gradient masking: Since most existing attacks rely on the gradient information of the loss function, a few of works [9, 31] attempt to conceal the gradient of a model. Ross and Doshi-Velez [31] trained differentiable models by penalising the degree to which small changes in the inputs can alter model predictions. In other words, a small perturbation cannot change the outputs of a model easily. Papernot *et al.* [9] proposed *defensive distillation*, where the distilled model is trained with the smoothed labels produced by an existing trained DNN model. Their experiments illustrated that distillation can improve the robustness of a network against small perturbation. However, these methods significantly increase the training complexity and *defensive distillation* remains vulnerable to CW attacks as shown in [7].

2.2.2 Detecting AEs: Although the complete classification can improve the robustness of a model, the AE can still be generated on the new classification model. Therefore, instead of modifying the model, the approach of detecting AE before the model receives more and more attention:

Secondary classification-based detection: Tian *et al.* [18] recorded the predictions of AE and CE under rotation and then trained a detector to classify AE based on the prediction details. The detector shows a good performance on MNIST and CIFAR10, but it is not suitable for large images such as ImageNet. Furthermore, the detector cannot classify AE correctly, which are from different attacks other than the considered attacks. We think the detector is overfitting for small images and single rotation is not sufficient to take account of the change of the predictions on large images. Gong *et al.* [16] fed a DNN with both AE and CE and attempt to classify with the learned DNN. While they can achieve a 100% success rate on detecting the AE generated from MNIST, the DNNs lead to a high FPR for CIFAR10. Also, their model yields a poor performance on large images (in our experiments, the models' FPR is 1 on images from ImageNet). The results reveal that it might be impractical to train DNNs on AE and CE directly. Metzen *et al.* [32] proposed to detect AE by looking at the inner convolutional layers of the network. They trained an extra detection neural network with outputs from the intermediate layers of the original neural network. The detection network showed good performance when the training and testing AE were generated from the same process and the perturbation was large enough. However, it cannot generalise well across different attack parameters and attack generation processes.

Inconsistency between AE and CE: Several studies [14, 15, 20] try to find the inconsistencies between AE and CE under certain scenarios, such as a *statistical test* using maximum mean discrepancy [20], higher weights on larger principal components after PCA [15] and different distribution under *kernel density estimation* or *Bayesian uncertainty estimates* [14]. However, they are only proven effective on one or two small datasets [22]. Meng and Chen [19] proposed MagNet, which combines a detector with a reformer, to detect abnormal AE. The detector discards the input far from CE, and the reformer moves AEs toward the manifold of normal examples. MagNet needs a threshold trained on the original images to detect AE, and it creates a large number of autoencoders as candidate detectors and reformers. Nevertheless, for CW attacks, this defence is unable to perform well even when the adversary is not attempting to evade it. Moreover, MagNet cannot work well on large images, which we will show in our experiments. Liang *et al.* [33] introduced an adaptive noise reduction method to detect AE. Specifically, scalar quantisation and spatial smoothing filters are used to reduce the effect of AE. We find that few of larger images and AE with low confidence are used in their paper to evaluate the defence. In addition, the detection rate under the CW white-box attack is very low (33%) even on the small dataset (MNIST). The most similar one to our work is feature squeezing, which utilises several squeezing skills, such as *bit-depth reduction*, *median smoothing,* and *non-local means*, to find the inconsistencies of AE and CE. Then through the inconsistency, they explore a threshold that can better differentiate AE and CE. In this paper, we will show a single threshold cannot handle different types of AEs from different attacks, and it is insufficient to detect AE based on the inconsistency of probability only.

3 Problem definition

3.1 Threat model

We assume that the attacker knows all the details of the classifier f, which means f is white-box to the attacker. Let D be a detector of defences. Depending on the knowledge of D the attacker knows, we consider two kinds of attacks:

Zero-knowledge adversary: The attacker does not know the existence of D and generates AE on f directly.

Adaptive adversary: The attacker knows the existence of D, owns the outputs of D, and attempts to evade both f and D simultaneously.

3.2 Detection evaluation metric

To evaluate the effectiveness of D, we decide to use the performance indicators of a machine-learning model, namely TP, FP, false negative (FN), and true negative (TN). D should ensure the accuracy of CEs and find AE as many as possible. Therefore, the FPR and TPR are used

$$\mathrm{FPR} = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN}) \tag{6}$$

$$\mathrm{TPR} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \tag{7}$$

A good detector should have a high TPR, but a low FPR, which means it labels AE with high accuracy and incorrectly labels CE as little as possible.

3.3 Evaluation process

Carlini and Wagner [22] proposed several recommendations for researchers to study new defences. We follow all of them in this paper:

Evaluate using a strong attack: In this paper, strong iterative attacks (CW, DeepFool, and IGSM) and non-iterative attacks (FGSM) are all used to evaluate our defence. Besides, we consider targeted and untargeted attacks for CW attacks, which are more powerful than other attacks.

Demonstrate white-box attacks fail: Evaluating defence with black-box attacks only is not enough. We also evaluate the scenario, where the attacker knows the defence and launch adaptive attacks to evade detection. As shown in [22], we construct a new model, which combines the classifier and the detector and then apply the attacks to this new model.

Report FPR and TPR: A good detector should achieve a high TPR first, which means it will find AE as many as possible. At the same time, we should ensure the low FPR, which means the detector misclassifies the CEs as few as possible.

Evaluate on a harder dataset: According to previous studies, attacking large images is easier than small images, which means defending against attacks on large images is harder than on small images. Previous detectors can detect most AE on small images, but not comprehensively being evaluated on large images. In this

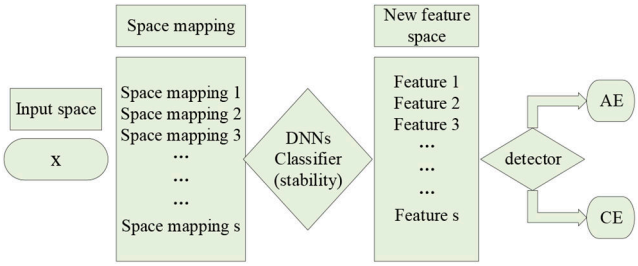

Fig. 1 *Whole framework of our design*

Fig. 2 *Rotation of different angles on an image*

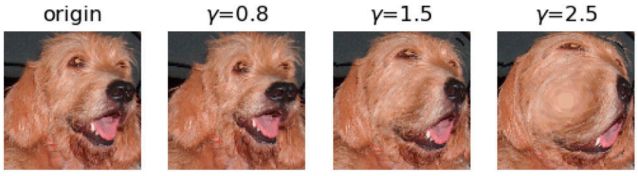

Fig. 3 *Fisheye of different factors on an image*

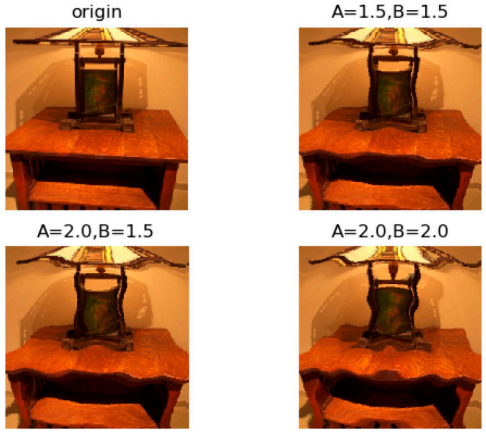

Fig. 4 *Original image and their wave versions*

paper, we focus on the high-resolution images in the real world. We look forward to a detector that can detect AE on high-resolution images with low errors, and it can be applied in the real world.

4 Method design

4.1 Overview

Owing to the complexity of AE, it is hard to distinguish AE from clean inputs directly. However, a phenomenon is observed that the stability of AE is poorer than CE. In other words, AE is sensitive to the preprocessing and the prediction of AE changed easily after the little modification on it. We think that a preprocessing can be considered as a linear or non-linear mapping of the input. Furthermore, for AE and CE, we suppose there is a big difference in behaviour under a set of linear or non-linear space mappings. In this paper, we take the detection of AE as a binary classification problem, as shown in Fig. 1. First, we adopt different mapping methods to preprocess these inputs. Then, we take the classification stability before and after image preprocessing as features in the new *feature space*. Finally, we train a detector to distinguish AE and CE based on these classification stability features.

More formally, we assume the original input $x_i \in \mathbb{R}^m (i = 1, 2, \ldots, n)$, the input space is $\mathcal{X}$, and the new space is $\mathcal{H}$. Let ϕ be a linear or non-linear transformation, and then there is a map

$$\phi(x): \mathcal{X} \to \mathcal{H} \tag{8}$$

Let ϕ_S be a set of mapping methods, namely

$$\phi_S = \{\phi_1, \phi_2, \ldots, \phi_s\} \tag{9}$$

In the space $\mathcal{H}$, we obtain the new representation of an input

$$x_i \to \phi_S(x_i) = \{\phi_1(x_i), \phi_2(x_i), \ldots, \phi_s(x_i)\} \tag{10}$$

The predictions are mainly composed of labels and probabilities. Thus, we should evaluate the changes of both labels and probabilities. For labels, let $E(x_1, x_2)$ measure the change, and then

$$E(x_1, x_2) = \text{equal}(C(x_1), C(x_2)) \tag{11}$$

where $C(x)$ is the classification of f on x and $E(x_1, x_2)$ is 1 if $C(x_1)$ equals to $C(x_2)$; otherwise, $E(x_1, x_2)$ is 0. For probabilities, let $L(x_1, x_2)$ measure the change, and then

$$L(x_1, x_2) = L(\boldsymbol{f}(x_1), \boldsymbol{f}(x_2)) \tag{12}$$

where $\boldsymbol{f}(x_1)$ and $\boldsymbol{f}(x_2)$ are n-dimensional vectors ($n = 1000$) of outputs and $L(\cdot)$ is an algorithm that measures the distance between the two distributions. Finally, we can get the features of input in the new space $\mathcal{H}$

$$\begin{aligned} \boldsymbol{F}_{(x_i)} = (&E(x_i, \phi_1(x_i)), L(x_i, \phi_1(x_i)), \ldots, \\ &E(x_i, \phi_s(x_i)), L(x_i, \phi_s(x_i))) \end{aligned} \tag{13}$$

Actually, after we get the distributions in the new space, we can observe the difference based on the trace distributions. Thus, the detection of AE is transformed into the binary classifications of AE and CE in the new space. Now, the key to detecting AE is to find better space mapping methods. In the following section, we will show how to choose better mapping methods.

4.2 Space mapping

Since there are lots of mapping methods for image processing, we decide to consider the space mapping methods from the following view: spatial location transformation, smoothing and hue, saturation, and lightness (HSL) colour modification. Then, we try to find representative mapping methods in these fields. Actually, any mapping method that can make the behaviours of AE and CE different can be considered. Our detector is *not limited* to the listed mapping methods and is also suitable for other proper mapping methods that have been tested experimentally.

4.2.1 Spatial location transformation: Intuitively, the target label of AE will change easily as the image's pixels change. If we perturb the spatial location of AE, the AE may not remain adversarial. Thus, different spatial transformation techniques are considered by us:

Rotation: Rotating the input can change the location distribution and make the decision of the model changed since neural networks cannot behave the same as a human. However, rotation also modifies on CE. How to rotate the input is an optimisation problem, and an example of rotation is shown in Fig. 2.

Flip: Flipping is also a kind of spatial transformation. Flipping an image horizontally [left to right (LR)] and flipping an image vertically [top to bottom (TB)] are the two flip skills used in our design.

Fisheye: Fisheye is originally used in the lens of a camera. Through fisheye, it is easy for us to observe the micro and macro features since fisheye can scale the local regions of images. Specifically, let w, h be the width and height of an image,

respectively. Here, (p_x, p_y) represents a pixel location in the image. In this paper, $(0 \le p_x, p_y \le 224)$. Then, let $(p_{x\text{centre}}, p_{y\text{centre}})$ be the central point of the image, r_{ori} be the original distance from the centre, and r_{new} be the new distance from the centre. We have

$$r_{\text{ori}} = \sqrt{(p_x - p_{x\text{centre}})^2 + (p_y - p_{y\text{centre}})^2} \tag{14}$$

$$r_{\text{new}} = \left(\frac{r_{\text{ori}}}{R}\right)^{\gamma} \times R \tag{15}$$

$$\begin{cases} p_{x\text{new}} = r_{\text{new}} \times \cos\theta + p_{x\text{centre}} \\ p_{y\text{new}} = r_{\text{new}} \times \sin\theta + p_{y\text{centre}} \end{cases} \tag{16}$$

where $\theta = \arctan(p_y - p_{x\text{centre}})/(p_x - p_{y\text{centre}})$, $R = \max(w, h)/2$, and γ is the perturbation factor. Then, we replace the pixel value at (p_x, p_y) with $(p_{x\text{new}}, p_{y\text{new}})$. When γ is bigger than 1, the image is concave; otherwise, it is convex. By the above coordinate transformation, we get a new image such as a fisheye. The examples are shown in Fig. 3.

Wave: Wave is a technique that can make the image look like a wave. Similar to fisheye, wave computes the new location of images. Instead of using (15), wave adopts the following computing:

$$r_{\text{new}} = r_{\text{ori}} + A \times \sin(B \times r_{\text{ori}}) \tag{17}$$

where A, B are parameters of wave. The examples of wave are shown in Fig. 4.

4.2.2 Smoothing: In this section, we introduce the smoothing techniques used in our detector:

Mean blur: This is done by convolving the image with a normalised box filter. It simply takes the average of all the pixels under the kernel area and replaces the central element. Therefore, the size of the kernel area can be adjusted to see the difference.

Bilateral blur: It is an advanced version of the Gaussian filter, which introduces another weight that represents how two pixels can be close (or similar) to one another in value. By considering both weights in the image, the bilateral filter can keep edges sharp while blurring the image.

Motion blur: Motion blur is the apparent streaking of moving objects in a photograph or a sequence of frames, such as a film or animation. Actually, it is a filter that can capture the motion state of an object. Through motion blur, we want to make the image blur with a different state.

Twirl blur: Twirl blur is a blur of the rotation state. Specifically, we first compute the angle θ and r_{ori} as shown in fisheye. Then, we get the angles set

$$\theta_{\text{new}} = \{\theta + i/100 | i = 1, 2, \ldots, \text{num}\} \tag{18}$$

where num are the parameters of twirl blur and can control the degree of twirl. Next, we compute the new position coordinates $(p_{x\text{new}}, p_{y\text{new}})$

$$\begin{cases} p_{x\text{new}} = r_{\text{ori}} \times \cos(\theta_{\text{new}}) + p_{x\text{centre}} \\ p_{y\text{new}} = r_{\text{ori}} \times \sin(\theta_{\text{new}}) + p_{y\text{centre}} \end{cases} \tag{19}$$

Actually, we get a set of new points, which are called the neighbourhood of the original location. Finally, we replace the pixel value at the original location (p_x, p_y) with the mean value of the set.

4.2.3 HSL colour modification: In this section, we also consider the modification of HSL colour. Hue is an attribute of a visual sensation according to which an area appears to be similar to one of the perceived colours: red, yellow, green, and blue, or to a combination of two of them. Saturation is the colourfulness of a stimulus relative to its own brightness. Lightness is the brightness relative to the brightness of a similarly illuminated white. Through the transformation in different colour spaces, we observe the difference between AE and CE.

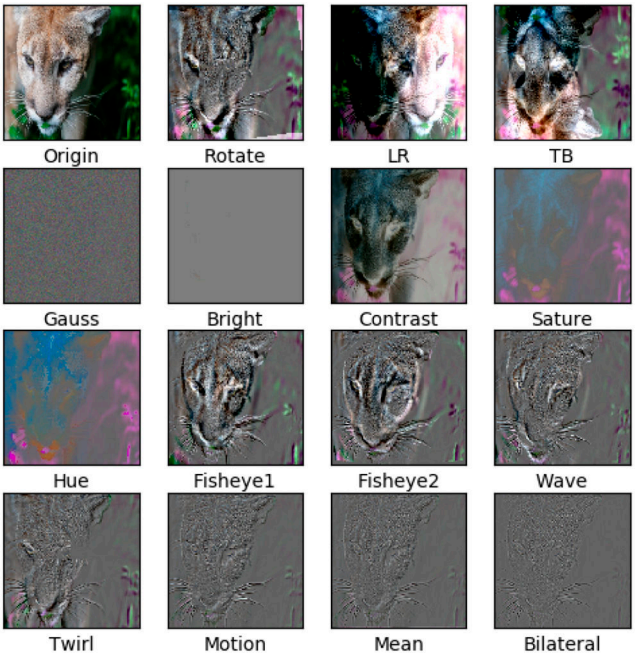

Fig. 5 *Visualisation of differences. We first transform the images with different mapping methods and then compute the differences. For better visual effects, we add all the differences to 100. Fisheye 1 and Fisheye 2 represent different γ values*

4.2.4 Other transformation: *Contrast* is the difference in luminance or colour that makes an object (or its representation in an image or display) distinguishable. *Gaussian noise* is the statistical noise with a probability density function equal to the normal distribution. Then, the noise is added to the original image. Finally, the image is clipped into the range of [−0.5, 0.5].

4.2.5 Analysis of mapping: To understand the impact of the mapping methods, we visualise the differences before and after the mapping in Fig. 5. We can clearly observe that different mapping methods transform images from different angles, such as the change of space location, the change of colour channels, and different levels of smoothing. This meets our requirements, namely we explore various ways in which images can be changed.

Besides, we want to know whether partial methods of the methods work well and whether they are complementary. We first choose the 10,000 AE from CW targeted and untargeted attacks (κ = 0, 10, 20, 30) as the test set. Then, we record the change of labels before and after the mapping methods. The results are shown in Table 1 and the numbers in this table represent the ratio of label changes. Although one single mapping method cannot achieve a high ratio among all AE, every mapping can make AE different to a certain extent. More importantly, the influence of labels on CE is far smaller than AE. Therefore, we can differentiate the AE from CE well through these mapping methods, and we will verify our design using more experiments in the next section.

4.2.6 Implementation: For the space mapping methods, we implement them with the TensorFlow. All of the smoothing skills can be implemented in the OpenCV library, except twirl blur. By using TensorFlow, we get the features of 10,000 images in <200 s. For the detector, we select the random forest as the classifier in the new space. Although many machine-learning models can be used, we find it is sufficient to achieve excellent performance with the random forest algorithm.

5 Experiment and evaluation

The previous section shows that different space mapping methods, as used to transform the images, have various effects that can

Table 1 Ratio of label changes among the mapping methods

Mapping	CE	Untargeted				Targeted	
CW	-	$\kappa = 0$	$\kappa = 10$	$\kappa = 20$	$\kappa = 30$	LL	Next
rotation	0.12	0.92	0.81	0.78	0.75	0.99	0.92
flip LR	0.04	0.95	0.81	0.75	0.69	0.99	0.95
flip TB	0.42	0.90	0.86	0.85	0.83	0.99	0.90
fisheye	0.06	0.93	0.70	0.58	0.51	1.00	0.91
wave	0.12	0.92	0.78	0.73	0.68	1.00	0.93
mean blur	0.18	0.90	0.87	0.86	0.86	1.00	0.90
motion blur	0.14	0.91	0.83	0.81	0.79	1.00	0.91
bilateral blur	0.16	0.90	0.84	0.83	0.82	1.00	0.90
twirl blur	0.10	0.92	0.75	0.68	0.65	1.00	0.91
hue	0.38	0.89	0.77	0.73	0.70	0.99	0.89
sature	0.24	0.85	0.64	0.55	0.49	0.99	0.85
lightness	0.02	0.72	0.02	0.01	0.01	0.98	0.72
Gauss Noise	0.12	0.91	0.82	0.79	0.75	0.99	0.92
contrast	0.25	0.88	0.79	0.76	0.73	0.99	0.88

Table 2 Evaluation on attacks

	Attack	Parameter	Success, %	Confidence	L_2
untargeted	FGSM [5]	$\epsilon = 0.06$	91.9	0.51	13.72
	IGSM [27]	$\epsilon = 0.06$	99.9	0.99	4.31
	DeepFool [6]	—	98	0.42	0.13
	CW [7]	$\kappa = 0$	100	0.45	0.21
		$\kappa = 10$	100	0.97	0.31
		$\kappa = 20$	100	0.99	0.43
		$\kappa = 30$	100	0.99	0.57
		$\kappa = 40$	100	0.99	0.69
targeted	CW [7]	LL	100	0.11	0.41
		next	100	0.44	0.20
	TGSM [27]	LL	94.9	0.90	4.41
	MI-FGSM [26]	$\epsilon = 0.3$	99.6	0.99	33.25
	EAD [34]	LL	99.9	0.32	18.93

modify the image to varying degrees. The mapping is expected to satisfy two properties: under the mapping: (i) the prediction of AE should change as much as possible (TP) and (ii) prediction of CE should change as little as possible (FP). In the following, we first examine the performance of different mapping methods based on the change of label and probabilities, since they are two metrics to measure the predictions. Then, we train a binary classifier in the new space to distinguish AE from CE, which is viewed as a detector. Finally, we evaluate the detector against represent attacks:

Dataset: We select 10,000 images from the ILSVRC2012 validation set and ensure that these images are predicted correctly (Top-1) by the ResNet-50 model. Besides, this subset covers all the 1000 classes of ImageNet. All the images are resized to 224×224×3 and normalised to [−0.5, 0.5], which are for adapting to the ResNet-50 model used later. The chosen 10,000 images consist of our CE set.

Model: We adopt the pre-trained ResNet-50 model from Github. Its initial accuracy on CE we selected is 100%.

Attacks: To measure the performance of the detector, we adopt both untargeted and targeted attacks. For targeted attacks, the representative CW targeted attacks [7], TGSM [27], MI-FGSM [26], and elastic-net attacks to DNNs (EAD) [34] are used for evaluation. For untargeted attacks, CW untargeted attacks [7], FGSM [5], IGSM [27], and DeepFool [6] are used for the test. We evaluate the classifier versus attacks and summarise the results in Table 2. Especially, the target-next attack means the target label is the second class in the original prediction and target-least likely (LL) attack means the target label is the least likely class in the original prediction. Confidence means the confidence level of the successful AE. As shown in Table 2, CW attacks can achieve high success rates both in target and untargeted attacks. For FGSM and IGSM attacks, we choose the parameters, which can cause high success rates. Obviously, targeted attacks are harder than untargeted attacks.

Table 3 Parameters to test

Mapping	Parameters
rotation	θ: 2, 3, 5, 7, 15, 30, 45, …, 150, 165
flip	LR, TB
fisheye	γ: 0.5, 0.6, …, 1.4, 1.5
wave	A = 1, 1.5, 2, 2.5; B = 1, 1.5, 2, 2.5
mean blur	kernel: 3×3, 4×4, 5×5, 6×6
motion blur	kernel:3×3, 4×4, 5×5, 6×6
bilateral blur	kernel:5×5, 7×7, 9×9
twirl blur	Num: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
hue	factor: 0.1, 0.2, …, 0.9, 1.0
sature	factor: 0.1, 0.3, …, 1.7, 1.9
lightness	ε: 0.01, 0.02, …, 0.09, 0.1
Gauss noise	σ: 0.02, 0.04, 0.06, 0.08, 0.1
contrast	factor: 0.3, 0.4, …, 1.1, 1.2

5.1 Sensitivity of labels

We expect that the label of a prediction of AE will change after the space transformation while not for CE since AE is expected to have a bigger sensitivity than CE. For each mapping method, different parameters are set to observe the prediction difference between AE and CE. The detail parameters are listed in Table 3 and unreasonable parameters are excluded in the experiment due to the serious damage to the original images. During the experiment, the

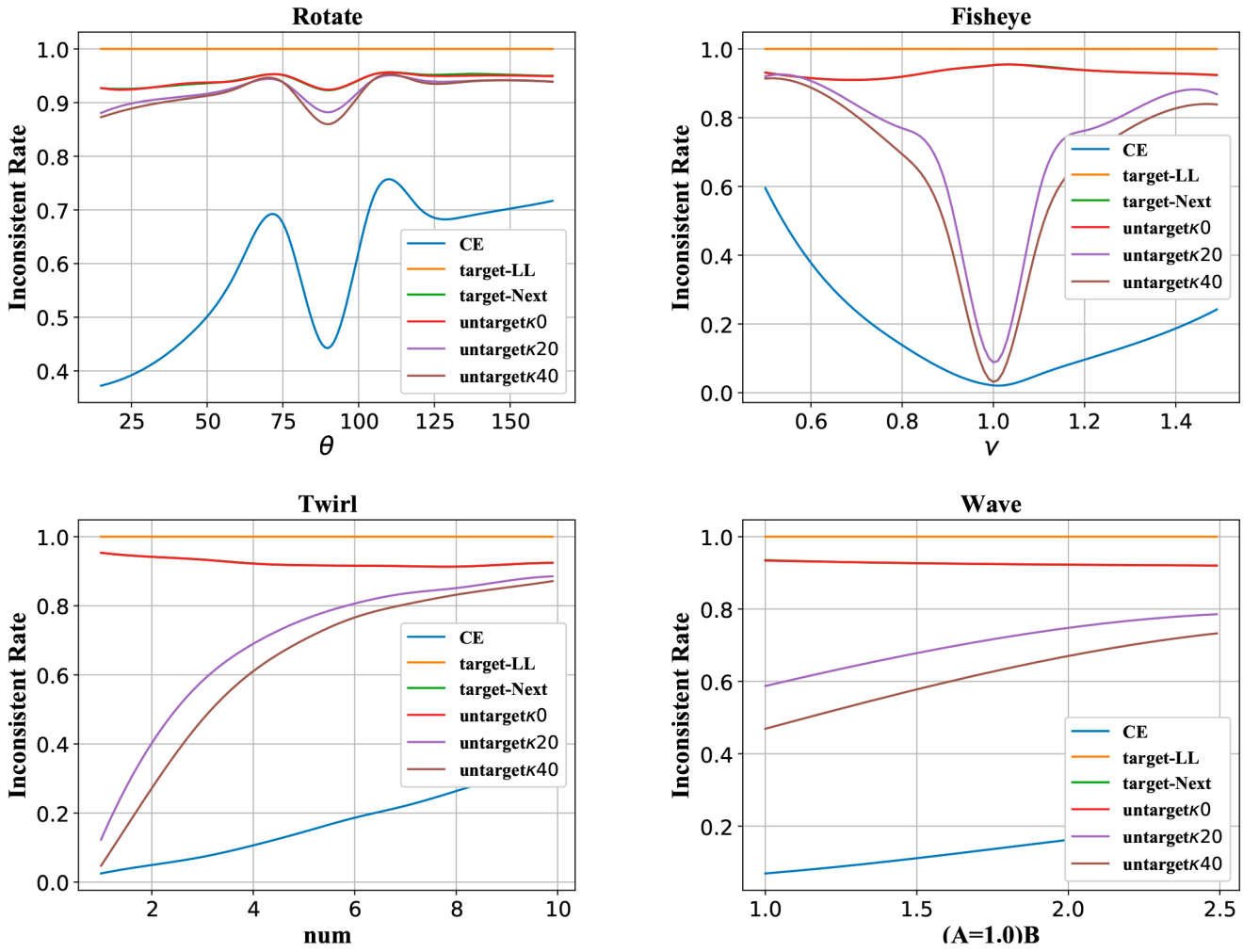

Fig. 6 *ICR of labels on different mapping methods. We record the rate at which labels are inconsistent after mapping*

Table 4 Adversarial class in the original prediction

Parameters	Rank 2, %	Rank 3, %	Rank 4, %	Rank 5, %
untarget-$\kappa = 0$	93.4	3.6	1.3	0.58
untarget-$\kappa = 10$	77.4	9.8	3.6	2.2
untarget-$\kappa = 20$	79.1	6.6	3.1	2.1
untarget-$\kappa = 30$	80.7	5.0	2.4	1.7
untarget-$\kappa = 40$	81.1	4.2	2.1	1.6

Table 5 Specific selected parameters

rotation: 2, 3, 4, 5, 7	flip: LR, TB
fisheye: $\gamma = 1.2, 0.8$	wave: ($A = 1$, $B = 1.5$)
mean blur kernel: 4×4	motion blur kernel:4×4
bilateral blur kernel: 5×5	twirl blur num: 5
hue factor = 0.5	sature factor = 0.1
lightness $\varepsilon = 0.1$	Gauss noise $\sigma = 0.04$
contrast	factor = 0.3

ratio of change on labels after transformation is recorded to measure the sensitivity of labels, which is also called *inconsistency rate* (ICR). Specifically

$$\mathrm{ICR} = \frac{1}{N}\sum_{i=1}^{N} E(C(x_i),\ C(\phi(x_i))) \tag{20}$$

where N is the number of test images, $C(x)$ is the label of x, and ϕ represents one space mapping method. $E(x_1, x_2)$ is 1 if $C(x_1)$ equals to $C(x_2)$; otherwise, $E(x_1, x_2)$ is 0. It should be noted that the ICR is computed on all testing data.

We show partial results in Fig. 6. On the basis of the behaviours of different mapping methods, we have the following findings:

- The ICRs of CW target-next attacks are almost the same as untarget-$\kappa = 0$'s. This is because CW untargeted attacks always push the class which is not the true class and its probability is the highest in original predictions to the adversarial class. Hence, most second-placed classes in original predictions appear on the adversarial classes of CW untargeted attacks. To verify this conjecture, we list the rank of adversarial classes within the top-5 predictions for the original image in Table 4. For all the untargeted attacks, the majority of adversarial classes appear in the top two of the original predictions.
- The ICRs of CE are lower than that of AE and are approximated to 0 under most mapping methods, such as smoothing techniques, wave, fisheye, lightness, and flip LR. Rotation with large angles contributes to the high ICR of CE for introducing more unrelated areas, while small angles not.
- The ICRs of CW target-LL attacks are near to 1, which means the sensitivity of targeted AE is very big. We also tested targeted AE with higher confidence ($\kappa = 10, 20, 30, 40$) and found that all of them behave very similarly, namely close to 1.
- Different from the targeted attacks, the ICR of untargeted AE goes down as the confidence goes up. Even by smoothing techniques, the ICRs of CE are below 0.8 under higher confidence ($\kappa = 20, 40$).
- Generally, smoothing techniques are better than spatial location transformation at distinguishing AE from CE, followed by HSL colour modification. We think spatial location transformation has a big effect on both AE and CE, which may make the labels of AE and CE change at the same time. Smoothing has a soft effect on the image and it is sufficient to make the label of AE change, but CE not. Besides, HSL colour modification creates

little in the model's decision making when handling AE with low confidence.

As shown in Fig. 6, the sensitivity of labels varies with the parameters. To choose a good parameter, we use the ICR to measure different parameters. Specifically, let $\mathrm{ICR}_{\mathrm{CE}}$ and $\mathrm{ICR}_{\mathrm{AE}}$ be the ICRs of CE and AE, respectively. Then, the performance of each parameter is P_{para}

$$P_{\mathrm{para}} = \frac{1}{5} \times \sum_{\kappa=0}^{40} \left(\mathrm{ICR}_{\mathrm{AE}_\kappa} - \mathrm{ICR}_{\mathrm{CE}}\right) \tag{21}$$

where κ is the confidence level used in CW attacks. Since the ICR s of target LL are nearly 1 and ICRs of target next are similar to untarget-$\kappa = 0$, we exclude the targeted attacks from P_{para}. Thus, we use the average result of AE under five κ values ($\kappa = 0$, 10, 20, 30, 40) as the final performance. The bigger the P_{para} value is, the better the ICR between CE and AE. As shown in Fig. 6, all the parameters in Table 3 are chosen for tests. Then, we use (21) to compute P_{para}s under different parameters. By selecting the one with the higher P_{para} value, we get 19 parameters of the mapping methods as shown in Table 5. We consider multiple parameters in rotation since we find that these angles have a different effect on AE. For fisheye, we choose two perturbation factors, which represent convex or concave fisheye. Next, we will use these specific mapping methods to measure the sensitivity of labels.

5.2 Sensitivity of probability distributions

Many algorithms can be used to compare the changes of probabilities, such as L_1 norm, L_2 norm, cosine distance, cross-entropy, Kullback–Leibler divergence, and Jensen–Shannon divergence. We have tried these methods and found that L_1 norm is the best. However, the performance difference between the different methods is not very great. Thus, we decide to adopt the L_1 norm as the metric in Section 4.1. Since L_1 distances are continuous values, we have to set different thresholds to select the parameters of mapping methods. To reduce the dependency on the dataset and the difficulty in defining thresholds, we use the mapping methods in Table 5 to observe the difference of the probability distributions. Although the selection of parameters does not consider the change of probability, we find that it is sufficient to differentiate the sensitivity of probability distributions.

To our surprise, the performance of the mapping methods is similar for each attack. However, the distinction is different for different types of attacks. For simplicity, we take *fisheye* ($\gamma = 0.9$) as an example and we plot the histogram of the L_1 distance among the probabilities. Since the probability ranges from 0 to 1, the L_1 norm between probabilities of before and after mapping ranges from 0 to 2. As shown in Fig. 7, the top subgraph of each image is the performance of AE and the bottom is CE. On the basis of the results of *fisheye* and other mapping methods, we have the following conclusions:

- The distribution of L_1 distance is different for different attacks. Therefore, a single L_1 distance as done in [17] cannot

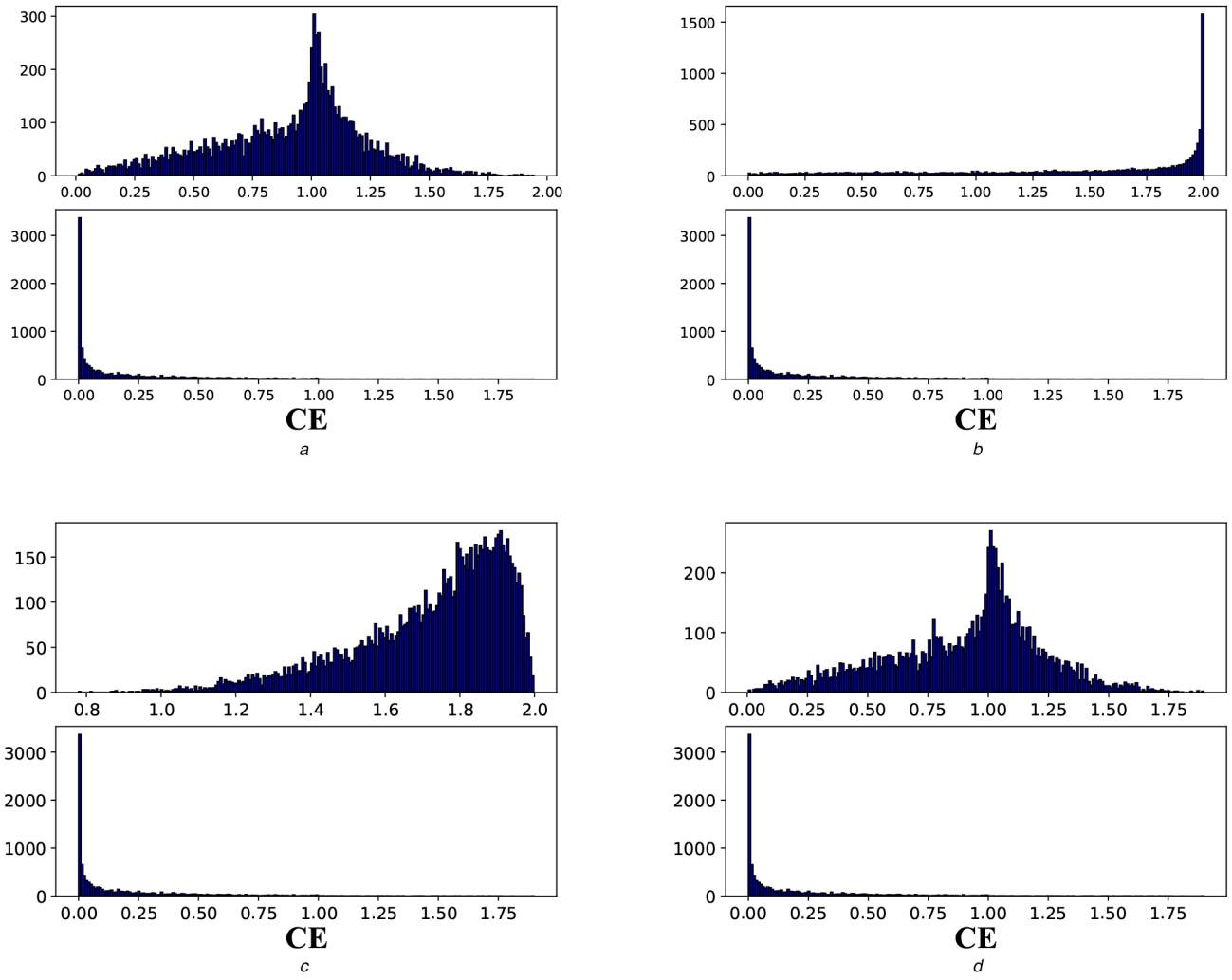

Fig. 7 *L_1 feature analyses under the mapping method – Fisheye. The top subgraph is the performance of AE and the bottom is CE. The range of L_1 is from 0 to 2. The horizontal axis represents L_1 distance and the vertical axis represents the number of images*
(a) Untarget-$\kappa = 0$, ***(b)*** Untarget-$\kappa = 10$, ***(c)*** Target LL, ***(d)*** Target next

differentiate all AE well, which might be generated from various adversarial attacks.

- The results of the targeted-next attack are similar to untarget-$\kappa = 0$. Besides, the L_1 distribution of FGSM and DeepFool attacks are similar to targeted attacks, not untargeted attacks with high confidence.
- The AE with high confidence shows a big difference from CE. Obviously, for AE with high confidence ($\kappa = 10, 20, 30$), most L_1 distances are near to 2, while CE not, which means the sensitivity of probability distribution for AE is very big under our mapping methods. Especially, for a target-LL attack, the intersection between CE and AE is the least among all attacks.

5.3 Detector

The sensitivities of AE and CE through different mapping methods can be viewed as a feature in the new space. For labels, the feature is 1 or 0, which means whether the labels are equal after mapping. For probabilities, the feature is the L_1 distance after each mapping. Since we want to know whether these features could distinguish AE from CE, we analyse several represented features in different dimensional spaces. Specifically, we choose *fisheye, wave, twirl, and motion blur* as the features and choose the AE from CW untargeted attacks to test.

First, we consider the one-dimensional (1D) space, namely only one feature will be tested. In Fig. 8, there is obviously a difference between AE and CE. However, we cannot distinguish all AE from CE under a single feature.

Then, we attempt to observe the distribution in the 2D space. Here, we display the change of probability since the label has two states only. As shown in Fig. 9, we divided the two features into a group to distribute AE and CE. In the 2D space, we find that the difference between the distribution of AE and CE is greater than that in 1D space.

Furthermore, we want to know whether the increase in dimensions can increase the degree of discrimination. Therefore, we display the partial distribution in the 3D space, as shown in Fig. 10. We treat each feature as a dimension, where both features of probability and label are considered. On the basis of the results, AE and CE are better differentiated in 3D space than lower dimensions.

Therefore, we assume that AE will reveal a different distribution from the CE in a high-dimensional space. Obliviously, this is a classification problem in the high-dimensional space, and we decide to use a machine-learning algorithm to construct the classifier, which is called the detector in our paper. Specifically, the 10,000 CE and corresponding AE are used to train and test our classifiers. We split 70% of the datasets into the training set and the rest into the test set. The 14,000 AE from CW untarget-$\kappa = 0$ and CW untarget-$\kappa = 10$ and 7000 CE make up the training set. The remaining are used to test the performance of our detector. For the machine-learning model, we select the typical model, random

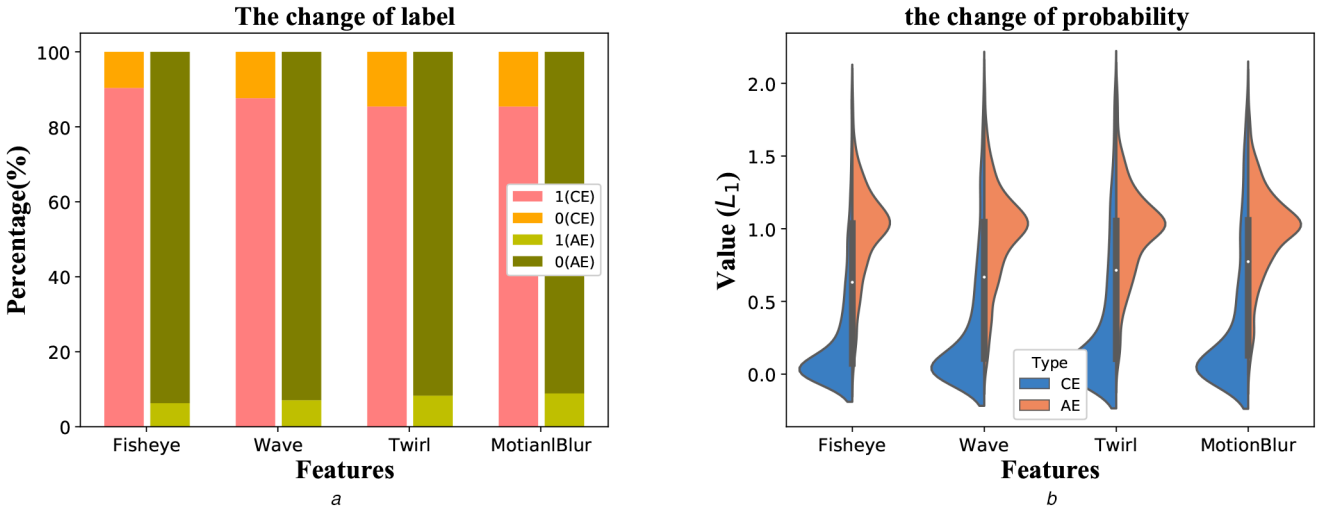

Fig. 8 *Distribution of AE and CE in the 1D space. The left represents the sensitivity of labels and the right shows the sensitivity of the probability distribution* ***(a)*** Change of label, ***(b)*** Change of probability

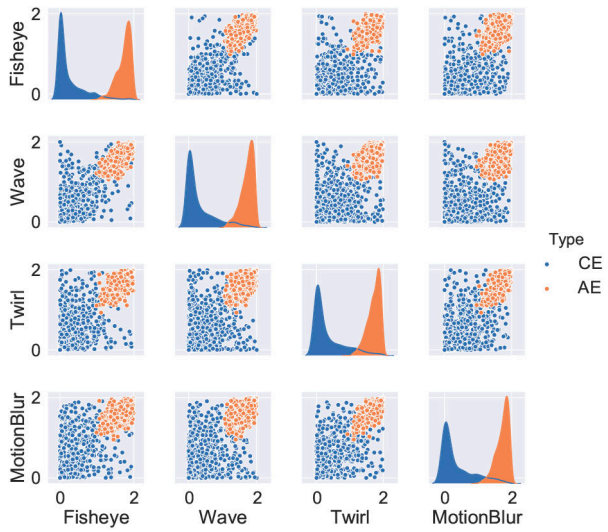

Fig. 9 *Distribution of AE and CE in the 2D space. The diagonal subgraphs represent the features themselves, while the non-diagonal subgraphs represent the 2D features*

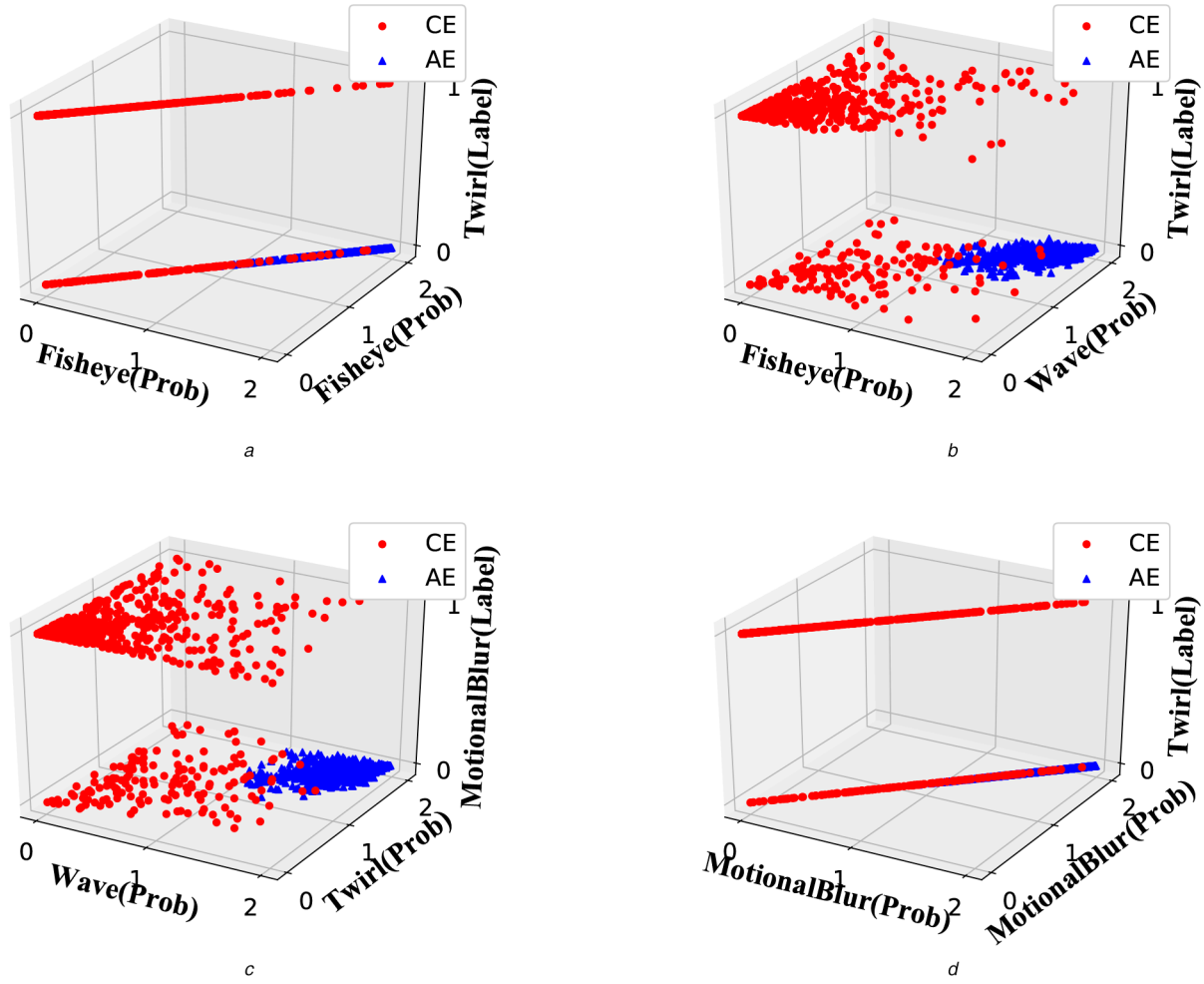

Fig. 10 *Distribution of AE and CE in the 3D space. We can clearly find the location of AE are significantly different from CE. The Fisheye and motional blur in (a) and (d) represent mapping methods with different parameters, respectively*

(a) The distribution of AE and CE in the 3D (Fisheye, Fisheye, Twirl) space, ***(b)*** The distribution of AE and CE in the 3D (Fisheye, Wave, Twirl) space, ***(c)*** The distribution of AE and CE in the 3D (Wave, Twirl, MotionalBlur) space, ***(d)*** The distribution of AE and CE in the 3D (MotionalBlur, MotionalBlur, Twirl) space

Table 6 Comparative experiments of different spatial mapping methods

Features	Fisheye + wave	Fisheye + wave + twirl	Fisheye + wave + twirl + motion blur	All
TPR	90.4%	93.1%	95.0%	99.1%
FPR	15.2%	13.0%	12.3%	0.3%

forest, to conduct our classification task. Through our experiments, it is verified that the random forest can carry out the task well.

To verify the importance of using all the image transformations for detection, we decide to use the features from several mapping methods, such as *fisheye, wave, twirl, and motion blur*. These mapping methods can distinguish AE from CE well, as shown in Fig. 6. Here, we choose all AE from CW attacks (targeted and untargeted) as the test set. The results of detections are shown in Table 6. From Table 6, the same model trained on features from partial mapping methods will lead to a high FPR. If we add more valid mapping methods, the model will achieve higher TPR with lower FPR. Therefore, we use all features from 19 mapping methods to build our detector.

The reason we combine both the label and probability is that these two make up the prediction. On the one hand, considering the changes of a label only will lead to high false positives since CE is also sensitive to the mapping methods to some extent. On the other hand, considering the changes in probability only is incomplete. For example, the adversarial attack may change the rank of classes in the prediction, but not change the series of probability values too much, which will not lead to a sufficiently big L_1 distance. In our following experiments, we indeed find the FPR is high (4%) when considering the changes of a label only. Besides, if the detector considers the changes in probability only, the TPR is poor when handling AE with larger perturbation, such as the AE generated by FGSM and IGSM. In the next section, we will show the overall performance of our detector on multiple test sets.

5.4 Evaluation of multiple attacks

5.4.1 Zero-knowledge adversary: To compare with other defences [17], we carry out their algorithms on the ImageNet dataset. For MagNet [19], which contains a detector and a reformer, we record images discarded by the detector and the images whose labels changed after the reformer. For feature squeezing, we first select the threshold according to the FPR of CE (<5%) and then adopt the best joint detection algorithm, which represents the best result of feature squeezing. The overall results of a zero-knowledge adversary are shown in Table 7, where we list the TPR in each attack column and the FPR in the last column. Besides, the average ROC–AUC score is 98.98%, which is also higher than [17] (94.24%).

On the basis of the results, we find that MagNet does not apply to the large high-resolution images, and the reformer will lead to a high FPR. We think the reason is that limited images cannot construct a high-precision reformer for large images and training the reformer with a huge amount of images is impractical. Unfortunately, feature squeezing also shows bad performance on

Table 7 Detection results of zero-knowledge adversary

Untargeted attacks							Targeted attacks					
Detections	CW $\kappa = 0$, %	CW $\kappa = 10$, %	CW $\kappa = 20$, %	DeepFool, %	FGSM, %	IGSM, %	CW next, %	CW LL, %	MI-FGSM, %	EAD, %	TGSM, %	FPR, %
MagNet [19]	71	65.5	58.2	84.9	56.3	35.3	84.9	83.4	77.6	70.3	45.4	17
feature squeezing [17]	54.6	82.5	75.8	49.8	42.3	63.1	55.1	99.9	67.7	99.7	38.8	4.9
our detector	**99.9**	**99.6**	**98.0**	**99.0**	**81.0**	**82.8**	**99.9**	**100**	**84.7**	**99.9**	**95.5**	**0.3**

The bold value represents the maximum value of each column, which means that our detector achieves highest detection rate on different adversarial attacks.

Table 8 Detection results of cross models

Model	CW untarget, %	CW target, %	DeepFool, %	FGSM, %	FPR, %	ROC–AUC, %
ResNet152	98.5	98.7	99.1	71.3	5.8	94.37
VGG16	96	96.3	97.1	61.6	3.6	95.71

Table 9 Success rate of an adaptive adversary

Detection	CW target next	CW target LL	CW target random	DeepFool
feature squeezing [17]	100%	100%	100%	57%
our detector	**41%**	**0**	**1.43%**	**4.6%**
L_2 (our detector)	1.44	0	2.76	0.15

most AE. Except for the CW targeted attacks, which are easy to detect, the TPRs for other attacks are all lower than our detector. Although our detector is trained on the CW untargeted attacks ($\kappa = 0$, 10) only, it can also detect other types of attacks with high TPR, such as CW targeted attacks, CW untargeted attacks ($\kappa = 20$, 30, 40), DeepFool, FGSM, and IGSM, which demonstrates its good robustness across attacks. Besides, the FPR is very low (<1%).

Cross models: We want to know whether the AE generated on other models can be detected by our detector. Thus, we select ResNet152 and VGG16 as the test models. The ResNet152 is the model family of ResNet50 and VGG16 is different from ResNet50 on architecture. After generating AE on these models, we evaluate the detector with the new AE and the results are shown in Table 8. For AE generated on VGG16, the TPR goes down a little bit; however, the TPR remains higher. The ROC–AUC scores are 94.37 and 95.71%, respectively. The results mean that the performance of our detector has not dropped significantly due to model changes. Furthermore, the attacker may use transferability [35] to attack the target model, namely they may use the AE generated by other models to fool the target model. However, this actually degenerates into a problem of detecting the AE of target models. We can use the detector deployed in the target model to recognise the AE as shown in Table 7.

5.4.2 Adaptive adversary: As an adaptive attacker, the adversary has complete knowledge of both the classifier and existence of our detector. Even the attackers know the mapping methods we used; they must generate AE and evade the image transformation simultaneously, which will lead to a decrease in AE's quality. Besides, it is hard for attackers to get useful image transformation methods since we can change the mapping methods and parameters. Thus, we simulate the adaptive adversary with the method proposed in [22]. In [22], our defence belongs to secondary classification-based detection. Specifically, we construct a new model G

$$G(x)_i = \begin{cases} Z_{\mathrm{F}}(x)_i & i \le N \\ Z_{\mathrm{D}}(x) * 2 * \max\left(Z_{\mathrm{F}}(x)\right) & i = N+1 \end{cases} \tag{22}$$

where Z_{D} and Z_{F} represent the output of the detector and logits of the classifier, respectively. Note that Z_{D} is the probability value of the AE. Thus, G acts as a classifier on $N+1$ classes, which combines the detector and classifier. If the detector views the input x as an AE, $Z_{\mathrm{D}}(x)$ will be larger than 0.5 and $G(x)$ will output the $N+1$ class. Otherwise, $G(x)$ will output the class as Z_{F} does. We then apply the adversarial attacks on the new model G. For targeted attack, if the input x is classified by G as the target class l ($l \le N$), then $C(x) = l$ and the detector D classifies x as a CE. We use the ASR to measure the effectiveness of adaptive attacks. Again, we compare the performance of the white-box attack with feature squeezing. The results are shown in Table 9. As shown in Table 9, feature squeezing is hard to defend against the CW attacks under the white-box setting. In contrast, our detector can resist all the AE from CW targeted-LL attack. Even for the CW target-random attack, the ASR is only 1.43%. When facing the easiest CW target-next attack, our detector significantly reduces the ASR from 100 to 41%. Actually, the CW target-random attack is the most representative attack since the target label cannot always be the next class. It is extremely difficult for an attacker to fool the model to output a specific label, but not the next class. When we attempt to generate AE with high confidence ($\kappa = 10$, 20), we surprisingly find that the ASR is dropped to 28%. Besides, the L_2 distance between AE and CE has a substantial increase compared with the AE without defence.

Randomisation: To counter the CW target-next attack in white-box scenarios, randomisation is considered by us [22]. After extracting the 38D features of the inputs, we randomly selected five features from the first iteration. Then, we replace five of 38 features with them in each iteration. Through this replacement process, we could increase the difficulty of the attack because there are always initial features exposed. The attacker cannot get the information of the five specific features and launch adaptive attacks to evade it. Given an adaptive adversary, we repeat the CW target-next attack on the detector with randomisation and find that the ASR drops to 3%. At the same time, we must remain the accuracy of the detector if randomisation is deployed. Thus, we retested the TPR with the modified detector given the adaptive adversary. As shown in Fig. 11, the performance of the randomised detector remains almost the same as that of the previous detector. Therefore, we believe our detector with randomisation can effectively hinder the adaptive adversary.

6 Discussion and limitation

Discussion: The biggest advantage of the proposed detector is its robustness. Different from previous works [16–18, 33], our detector works well across attack algorithms and models. Besides, our detector exhibits a lower FPR and higher TPR under white-box attacks than before. Although dozens of mapping methods are used in our detection, the overhead is very small (<0.1 s per image). Since the mapping methods used in our detector are sufficient to detect most AE, we did not consider other mapping methods, such as JPEG compression, resizing, shift, and padding. However, our defence is open, and other new mapping methods can be directly integrated into the detector easily. It should be noted that our detector is significantly different from ensemble methods [17, 25]. We want to explore the trajectory of AE under various mapping

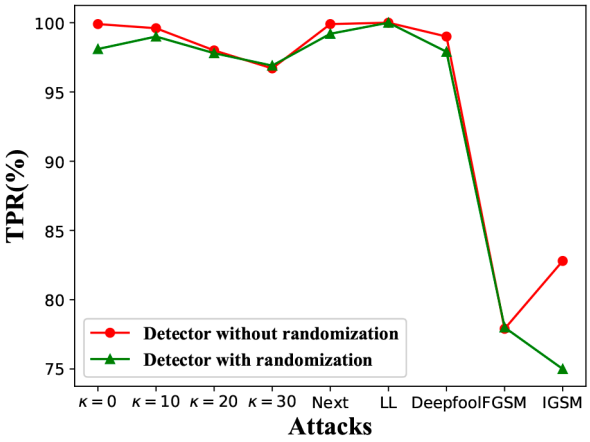

Fig. 11 *TPR between the original detector and detector with randomisation*

methods, not the decision based on multiple detectors. Besides, the attack algorithms in [22, 23], which can break ensemble detections easily, are implemented in this paper to evaluate our detector, and our detector demonstrates promising performance against them.

Limitation: Our detector depends on the difference between AE and CE, so we need to generate AE first. If getting AE is hard for some scenarios, the training will get blocked. Fortunately, only small batches of AE are used in training as shown in Section 5.3. The number of AE is far below the AE using in adversarial training. In addition, the TPR of white-box attacks is not very high yet without randomisation, and this leaves room for attackers who own details of the detector to launch an attack. As shown in Table 9, the AE of CW targeted-next attack is not well-detected. However, target labels of attackers are not always easy (the next class of the original label) and they might be far from the original label. Moreover, a randomised detector can block most white-box attacks while allowing a little bit of TPR loss.

7 Conclusions

Many works focus on the defences or detection of AE in DNNs. However, most of such existing methods are not attack independent and model independent. Besides, their detections only work on small images. In this paper, we propose an effective defence based on space mapping. The change of label and probability are both considered under different mapping methods. Through space mapping, we can trace the AE in the new space. Finally, we train a detector to learn the trace difference in the new space between AE and CE with machine-learning models. Our detector provides four important features: (i) can work well across different parameters of the same attack; (ii) can work well across different types of attacks; (iii) can work well across different models, and (iv) can be deployed directly without modifying the original models.

We evaluate our detector under both zero-knowledge adversary and adaptive adversary. The experiments demonstrate that our detector can achieve a high TPR in detecting the AE generated by different attack algorithms given zero-knowledge adversaries, and can effectively raise the bar for adaptive adversaries. Compared to the state-of-the-art defences, our detector performs better on both TPR and FPR. Our research demonstrates that AE and CE can be effectively differentiated with common space mapping methods. Furthermore, we will explore more differences between AE and CE to develop an effective defence that relies on train data as little as possible and achieves better performance under the adaptive adversary scenario.

8 Acknowledgments

This work was partly supported by the National Key Research and Development Programme of China (2018YFB2100404 and 2018YFB1800601), the NSFC under No. 61772466, U1936215, and U1836202, the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under no. LR19F020003, the Provincial Key Research and Development Programme of Zhejiang, China under No. 2018C03052, 2020C01021, and 2019C01055 and Major Scientific Project of Zhejiang Lab (2018FD0ZX01).

9 References

[1] Bojarski, M., Testa, D.D., Dworakowski, D., *et al.*: 'End-to-end learning for self-driving cars', CoRR, arXiv:1604.07316, 2016

[2] Cireşan, D., Meier, U., Masci, J., *et al.*: 'Multi-column deep neural network for traffic sign classification', *Neural Netw.*, 2012, **32**, pp. 333–338

[3] Szegedy, C., Zaremba, W., Sutskever, I., *et al.*: 'Intriguing properties of neural networks'. Second Int. Conf. Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April 2014, Conference Track Proceedings

[4] Papernot, N., McDaniel, P.D., Jha, S., *et al.*: 'The limitations of deep learning in adversarial settings'. IEEE European Symp. Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, 21–24 March 2016, pp. 372–387

[5] Goodfellow, I.J., Shlens, J., Szegedy, C.: 'Explaining and harnessing adversarial examples'. Third Int. Conf. Learning Representations ICLR 2015, San Diego, CA, USA, 7–9 May 2015, Conference Track Proceedings

[6] Moosavi-Dezfooli, S., Fawzi, A., Frossard, P.: 'DeepFool: a simple and accurate method to fool deep neural networks'. 2016 IEEE Conf. Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016, pp. 2574–2582

[7] Carlini, N., Wagner, D.A.: 'Towards evaluating the robustness of neural networks'. 2017 IEEE Symp. Security and Privacy SP 2017, San Jose, CA, USA, 22–26 May 2017, pp. 39–57

[8] Bhagoji, A.N., Cullina, D., Mittal, P.: 'Dimensionality reduction as a defense against evasion attacks on machine-learning classifiers', CoRR, arXiv:1704.02654, 2017

[9] Papernot, N., McDaniel, P.D., Wu, X., *et al.*: 'Distillation as a defense to adversarial perturbations against deep neural networks'. IEEE Symp. Security and Privacy SP 2016, San Jose, CA, USA, 22–26 May 2016, pp. 582–597

[10] Zheng, S., Song, Y., Leung, T., *et al.*: 'Improving the robustness of deep neural networks via stability training'. 2016 IEEE Conf. Computer Vision and Pattern Recognition CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016, pp. 4480–4488

[11] Prakash, A., Moran, N., Garber, S., *et al.*: 'Deflecting adversarial attacks with pixel deflection'. 2018 IEEE Conf. Computer Vision and Pattern Recognition CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018, pp. 8571–8580

[12] Xie, C., Wang, J., Zhang, Z., *et al.*: 'Mitigating adversarial effects through randomization'. Sixth Int. Conf. Learning Representations ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018, Conference Track Proceedings

[13] Guo, C., Rana, M., Cissé, M., *et al.*: 'Countering adversarial images using input transformations'. Sixth Int. Conf. Learning Representations ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018, Conference Track Proceedings

[14] Feinman, R., Curtin, R.R., Shintre, S., *et al.*: 'Detecting adversarial samples from artifacts', CoRR, arXiv:1703.00410, 2017

[15] Hendrycks, D., Gimpel, K.: 'Early methods for detecting adversarial images'. Fifth Int. Conf. Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017, Workshop Track Proceedings

[16] Gong, Z., Wang, W., Ku, W.S.: 'Adversarial and clean data are not twins', CoRR, arXiv:1704.04960, 2017

[17] Xu, W., Evans, D., Qi, Y.: 'Feature squeezing: detecting adversarial examples in deep neural networks'. 25th Annual Network and Distributed System Security Symp. NDSS 2018, 18–21 February 2018, pp. 2–8

[18] Tian, S., Yang, G., Cai, Y.: 'Detecting adversarial examples through image transformation'. Proc. 32nd AAAI Conf. Artificial Intelligence (AAAI-18), New Orleans, LA, USA, 2–7 February 2018, pp. 4139–4146

[19] Meng, D., Chen, H.: 'MagNet: a two-pronged defense against adversarial examples'. Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security CCS 2017, Dallas, TX, USA, 30 October–03 November 2017, pp. 135–147

[20] Grosse, K., Manoharan, P., Papernot, N., *et al.*: 'On the (statistical) detection of adversarial examples', CoRR, arXiv:1702.06280, 2017

[21] Madry, A., Makelov, A., Schmidt, L., *et al.*: 'Towards deep learning models resistant to adversarial attacks'. Sixth Int. Conf. Learning Representations ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018, Conference Track Proceedings

[22] Carlini, N., Wagner, D.: 'Adversarial examples are not easily detected: bypassing ten detection methods'. Proc. Tenth ACM Workshop on Artificial Intelligence and Security, 2017, pp. 3–14

[23] He, W., Wei, J., Chen, X., *et al.*: 'Adversarial example defense: ensembles of weak defenses are not strong'. 11th USENIX Workshop on Offensive Technologies WOOT 2017, Vancouver, BC, Canada, 14–15 August 2017

[24] Carlini, N., Wagner, D.: 'MagNet and 'efficient defenses against adversarial attacks' are not robust to adversarial examples', CoRR, arXiv:1711.08478, 2017

[25] Abbasi, M., Gagné, C.: 'Robustness to adversarial examples through an ensemble of specialists'. Fifth Int. Conf. Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017, Workshop Track Proceedings

[26] Dong, Y., Liao, F., Pang, T., *et al.*: 'Boosting adversarial attacks with momentum'. 2018 IEEE Conf. Computer Vision and Pattern Recognition CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018, pp. 9185–9193

[27] Kurakin, A., Goodfellow, I.J., Bengio, S.: 'Adversarial examples in the physical world'. Fifth Int. Conf. Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017, Workshop Track Proceedings

[28] Akhtar, N., Mian, A.S.: 'Threat of adversarial attacks on deep learning in computer vision: a survey', *IEEE Access*, 2018, **6**, pp. 14410–14430

[29] Papernot, N., McDaniel, P., Sinha, A., *et al.*: 'Towards the science of security and privacy in machine learning', CoRR, arXiv:1611.03814, 2016
[30] Dziugaite, G.K., Ghahramani, Z., Roy, D.M.: 'A study of the effect of JPG compression on adversarial images', CoRR, arXiv:1608.00853, 2016
[31] Ross, A.S., Doshi-Velez, F.: 'Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients'. Proc. 32nd AAAI Conf. Artificial Intelligence (AAAI-18), New Orleans, LA, USA, 2–7 February 2018, pp. 1660–1669
[32] Metzen, J.H., Genewein, T., Fischer, V., *et al.*: 'On detecting adversarial perturbations'. Fifth Int. Conf. Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings
[33] Liang, B., Li, H., Su, M., *et al.*: 'Detecting adversarial image examples in deep neural networks with adaptive noise reduction', *IEEE Trans. Dependable Secur. Comput.*, 2018. The Print ISSN: 1545-5971, Electronic ISSN: 1941-0018, CD: 2160-9209. p.1, https://ieeexplore.ieee.org/document/8482346
[34] Chen, P., Sharma, Y., Zhang, H., *et al.*: 'EAD: elastic-net attacks to deep neural networks via adversarial examples'. Proc. 32nd AAAI Conf. Artificial Intelligence (AAAI-18), New Orleans, LA, USA, 2–7 February 2018, pp. 10–17
[35] Liu, Y., Chen, X., Liu, C., *et al.*: 'Delving into transferable adversarial examples and black-box attacks'. Fifth Int. Conf. Learning Representations ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings