Principal Component Adversarial Example

Yonggang Zhang, Xinmei Tian, *Member, IEEE*, Ya Li, Xinchao Wang, and Dacheng Tao, *Fellow, IEEE*

***Abstract*— Despite having achieved excellent performance on various tasks, deep neural networks have been shown to be susceptible to adversarial examples, i.e., visual inputs crafted with structural imperceptible noise. To explain this phenomenon, previous works implicate the weak capability of the classification models and the difficulty of the classification tasks. These explanations appear to account for some of the empirical observations but lack deep insight into the intrinsic nature of adversarial examples, such as the generation method and transferability. Furthermore, previous works generate adversarial examples completely rely on a specific classifier (model). Consequently, the attack ability of adversarial examples is strongly dependent on the specific classifier. More importantly, adversarial examples cannot be generated without a trained classifier. In this paper, we raise a question: what is the real cause of the generation of adversarial examples? To answer this question, we propose a new concept, called the *adversarial region*, which explains the existence of adversarial examples as perturbations perpendicular to the tangent plane of the data manifold. This view yields a clear explanation of the transfer property across different models of adversarial examples. Moreover, with the notion of the adversarial region, we propose a novel target-free method to generate adversarial examples via principal component analysis. We verify our adversarial region hypothesis on a synthetic dataset and demonstrate through extensive experiments on real datasets that the adversarial examples generated by our method have competitive or even strong transferability compared with model-dependent adversarial example generating methods. Moreover, our experiment shows that the proposed method is more robust to defensive methods than previous methods.**

***Index Terms*— Deep learning, adversarial examples, classification, manifold learning.**

I. Introduction

DEEP neural networks (DNNs) are powerful machine learning models that have achieved state-of-the-art or even human-competitive performance on visual, speech and other tasks [1]–[7]. However, recent works have revealed a surprising discovery: state-of-the-art deep neural networks are vulnerable to adversarial examples, which are samples generated from real samples with carefully designed imperceptible perturbations [8]–[15]. Additionally, these adversarial examples usually have good transferability across different machine learning models [8]. For example, adversarial examples that easily fool neural networks can also cause failures in other models, such as support vector machines (SVMs), logistic regression, decision trees and other neural networks with different architectures. The existence of such adversarial examples demonstrates that neural networks have a different working mechanism than human version for pattern recognition tasks [16]. Adversarial examples also pose potential security threats for machine learning systems. The study of adversarial examples is therefore crucial to improve the robustness of existing learning models and to decrease the discrepancy between neural networks and human version.

Existing works explain the phenomenon of adversarial examples from two aspects. The first focuses on the attributes of neural networks, such as nonlinearity [16], the linear nature of DNNs [9], the flatness of decision boundaries [17] and the large local curvature of decision boundaries [18]. The second argues that the difficult nature of classification tasks leads to the existence of adversarial examples [13]. These explanations, however, often fall short in terms of generalization [19] and have their own drawbacks, among which the key is their failure to explain the transferability of adversarial examples. For example, adversarial examples generated from neural networks can be easily transferred to linear models such as linear SVMs and nonlinear models like logistic regression. In other cases, adversarial examples fool complicated neural networks but not simple models such as KNN.

In this paper, we propose a new perspective to explain the existence of adversarial examples. Considering the transferability of adversarial examples, it is reasonable to hypothesize that they are special perturbations residing in high-dimensional space. Specifically, the direction of the perturbation is the fastest one that deviates from the surface of the manifold, and the magnitude of the perturbation is sufficiently small. We regard the regions in which such special perturbations reside as *adversarial regions*. The notion of an adversarial region provides an intuitive understanding of the adversarial phenomena from a geometric perspective. For example, the transferability of adversarial examples is essentially a result of the fact that there is an intersection of the divisions of the adversarial region and different classifier boundaries. Moreover, the adversarial region makes it possible to utilize the data distribution instead of a classifier to generate adversarial

Manuscript received February 18, 2019; revised December 25, 2019; accepted February 9, 2020. Date of publication February 28, 2020; date of current version March 4, 2020. The work of Xinmei Tian was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1002203 and the NSFC under Grant 61872329 and Grant 61572451. The work of Xinchao Wang was supported in part by the Startup Funding of Stevens Institute of Technology. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Liang Wang. *(Corresponding author: Xinmei Tian.)*

Yonggang Zhang, Xinmei Tian, and Ya Li are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China (e-mail: yonggang@mail.ustc.edu.cn; xinmei@ustc.edu.cn; muziyiye@mail.ustc.edu.cn).

Xinchao Wang is with the Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: xinchao.wang@stevens.edu).

Dacheng Tao is with the UBTECH Sydney Artificial Intelligence Centre, The University of Sydney, Sydney, NSW 2008, Australia, and also with the Faculty of Engineering, School of Computer Science, The University of Sydney, Sydney, NSW 2008, Australia (e-mail: dacheng.tao@sydney.edu.au).

Digital Object Identifier 10.1109/TIP.2020.2975918

examples. Experiments are conducted on a synthetic dataset to verify the hypothesis of our proposed adversarial region.

The other contribution of this paper is that we propose a target-free adversarial example generation method based on the proposed adversarial region. The utilized concept "target-free" encodes the core idea of our method that adversarial examples are generated without the dependence on any classifiers. Compared with previous widely used model-dependent methods such as fast gradient sign (FGS) [9] and the state-of-the-art method CW [20], which utilize a well-trained classifier to generate adversarial examples, our proposed method is fully unsupervised. Adversarial examples generated by supervised generation methods rely heavily on the classifier, e.g., the quality of the adversarial examples can be reduced dramatically when insufficient labeled data samples are available for the classifier. In addition, adversarial examples generated by model-dependent methods can overfit the specific classifier [21], which could result in weak transferability. On the contrary, adversarial examples generated by the target-free method never suffer from overfitting, and insufficient labeled data has a limited influence. Hence, the target-free generation method is promising.

In a high-dimensional manifold, direct and explicit computation of the surface of the manifold is difficult, especially when the data distribution is complicated. Thus, we adopt principal component analysis (PCA) to approximate the principal directions of the manifold. Our approach, called principal component adversarial example (PCAE), is to the best of our knowledge the first target-free adversarial example generation method.

We compared our proposed PCAE with FGS and the state-of-art CW method on the MNIST, CIFAR-10 and ImageNet datasets. The experimental results demonstrate the effectiveness of our proposed method, especially when the target model is defensive.

II. Related Work

Adversarial examples have received considerable attention since they were first discussed [22]. Although neural networks can achieve state-of-the-art performance on extensive machine learning tasks, they are vulnerable to normal examples with structural imperceptible noise. This interesting phenomenon has inspired researchers to study the intrinsic properties of neural networks and design defensive methods to resist the attack of adversarial examples.

Substantial efforts have been made to explain the existence of adversarial examples. Szegedy et al. explained that the nonlinearity of neural networks caused the misclassification of adversarial examples [22]. They argued that regions containing no training examples could represent the same objects from different viewpoints because of the deep stack of nonlinear layers between the input and output units of a neural network. Goodfellow et al. provided a simpler explanation that the linearity of neural networks also could cause the misclassification of adversarial examples [9]. The output activation caused by imperceptible perturbation could grow linearly with the dimensionality of the input data. Consequently, the output could change significantly for high-dimensional problems, even if the perturbation was very small. However, Tanay and Griffin [23] identified several inadequacies of the linearity perspective. By contrast, Fawzi et al. proved a general upper bound on the robustness of classifiers and showed the discrepancy between the robustness of a classifier to adversarial perturbations and the robustness to random noise [13]. Additionally, the flatness of decision boundaries [17] and the large local curvature of the decision boundaries [18] provide more viewpoints on the existence of adversarial examples, but these viewpoints could be improper when they align with each other [19].

Constructing adversarial examples with high confidence is regarded as another key issue. Goodfellow et al. proposed a FGS method to generate imperceptible perturbation according to the direction of the loss gradient [9]. Kurakin et al. proposed a variant of the FGS method by applying FGS several times with a smaller step size [24]. Christian et al. proposed a box-constrained L-BFGS method to generate adversarial examples [22]. The state-of-the-art CW method creates quasi-imperceptible perturbations by restricting their l_2 norms, and it was shown that these adversarial examples transfer well. However, the efficiency of this method is not promising. All these methods rely on supervised training data or well-trained neural networks to generate adversarial examples. If the supervised training data are insufficient or the model is not trained well, the performance decreases dramatically. By contrast, our PCAE can generate adversarial examples according to the manifold of the data without any supervised information or classification model.

Several recent works [23], [25]–[28] considered adversarial examples from a geometrical perspective. Although these works are all based on the geometrical perspective, there are two types of methods. Some studies [25]–[27] argue that adversarial examples are related to the data manifold, whereas others highlight the geometric insights on the classifier's decision surface [28]. However, these methods [25]–[27] concentrate on the resistance to adversarial examples and do not explicitly provide the relationships between adversarial examples and the data manifold. Moreover, the generation methods derived from [28] are model-dependent. In [29], instead of finding target-free adversarial examples, SVD was applied to determine the direction of the perturbation generated by model-dependent methods.

III. Method

In this section, we start by presenting a brief review of previous models that relate the adversarial phenomenon to data manifolds. These methods show the sensibility of neural networks to some specific changes. This inspires us to propose a novel concept, called the *adversarial region*, to describe the adversarial phenomenon formally. We then explain the transferability of adversarial examples and devise a target-free method to generate adversarial examples using the proposed adversarial region based on PCA.

A. Motivation

A well-trained classifier should be invariant to small distortions of the input manifold, such as rotations, translations or

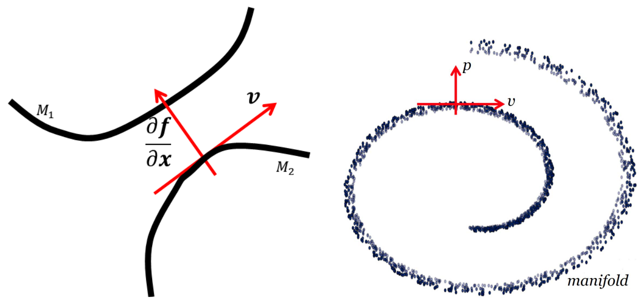


Fig. 1. (a) Shows the main idea of the tangent-prop algorithm. The figure illustrates two manifolds ($\boldsymbol{M}_1$ and $\boldsymbol{M}_2$), one per class, and $\boldsymbol{v}$ is the manifold tangent vector specifying how one can change the data point infinitesimally while staying on the manifold. One way to make the classifier invariant to the local distortion is to penalize the magnitude of the product of $\boldsymbol{v}$ and $\frac{\partial f}{\partial \boldsymbol{x}}$. (b) Shows the small variance along $\boldsymbol{p}$ that is orthogonal to the tangent plane.

small noise [30]–[32]. Such local invariance can be achieved by enforcing $\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$ to be orthogonal to the manifold tangent vectors $\boldsymbol{v}$ at point $\boldsymbol{x}$, where $\boldsymbol{x}$ is the input and $f(\boldsymbol{x})$ is the output of the classification model, such as a neural network. Note that the tangent vectors refer to the directions of the distortions in the manifold. The tangent-prop algorithm [30] is one typical method along this line, where the magnitude of the dot product of the directional derivative of f at $\boldsymbol{x}$ and all distortion directions $\boldsymbol{v}$ is penalized by $R = \sum_i \left\| \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \cdot \boldsymbol{v}_i \right\|^2$. This regularizer makes $f(\boldsymbol{x})$ insensitive to the tangent directions at point $\boldsymbol{x}$. In other words, the tangent-prop algorithm forces the direction of $\frac{\partial f}{\partial \boldsymbol{x}}$ to be orthogonal to the tangent vectors at point $\boldsymbol{x}$ to guarantee that the is model robust to small distortions. Figure 1(a) illustrates the main idea.

According to the conclusions of previous works [30], [32], [33], the output ($f(\boldsymbol{x})$ or the cost $J(\boldsymbol{x})$) of a well-trained classifier should be insensitive to the tangent vectors, i.e., $\frac{\partial f}{\partial \boldsymbol{x}}$ or $\nabla_{\boldsymbol{x}} J$ is orthogonal to the tangent plane at point $\boldsymbol{x}$. In most cases, this is reasonable because the probability density of the distortions is expected to fall off sharply as the direction of the distortion deviates from the tangent plane of the manifold [31], [34], see Figure 1(b). Although these types of distortions have a very low probability of being present, well-generalized models are highly sensitive to distortions distributed in the direction orthogonal to the tangent plane of the manifold at point $\boldsymbol{x}$.

Considering the low probability of natural adversarial examples (not generated by well-designed algorithms) and the conclusions of previous studies, it is reasonable to assume that the direction orthogonal to the tangent plane is the key to understanding adversarial examples.

B. Adversarial Region

In this section, we introduce the notion of the adversarial region to help us better understand the adversarial phenomenon. We first introduce the notation and basic definitions of the manifold and then present the proposed adversarial region.

As defined in [35], a d-dimensional manifold M is a set that is locally homeomorphic with $\mathcal{R}^d$. That is, for each $\boldsymbol{x} \in M$, there is an open neighborhood around $\boldsymbol{x}$, $N_{\boldsymbol{x}}$ and a homeomorphism $f : N_{\boldsymbol{x}} \to \mathcal{R}^d$. These neighborhoods are referred to as coordinate patches, and the map is referred to as a coordinate chart. The image of the coordinate charts is called the parameter space.

According to this definition, a homeomorphism refers to a continuous function whose inverse is also a continuous function. Intuitively, the data points on the manifold can be characterized locally by a low-dimensional vector. Such a low-dimensional representation for a particular example is called its embedding.

The premise of our adversarial region perspective is the manifold hypothesis. According to this hypothesis, real-world data presented in high-dimensional spaces are expected to concentrate in the vicinity of a manifold M of much lower dimensionality R^d embedded in R^D $(D > d)$. In other words, a d-dimensional manifold M passes through the D-dimensional data space. We use two functions to model this process: one describes the manifold, and the other provides a low-dimensional representation given a point. Data points lying on the manifold are the average of the points that have the same low-dimensional representation. In the following, we introduce these two functions formally.

Let $\boldsymbol{x} \in \mathcal{R}^D$ denote a point in a dataset. Let $\boldsymbol{F} : \mathcal{R}^d \to \mathcal{R}^D$ be a continuous function that takes $\boldsymbol{\lambda} \in \mathcal{R}^d$ as input and produces a point $\boldsymbol{F}(\boldsymbol{\lambda}) = (F_1(\boldsymbol{\lambda}), \ldots, F_D(\boldsymbol{\lambda}))$ that lies on the d-dimensional manifold. For any $\boldsymbol{x}$, let $\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x})$ denote the projection index, that is, the value $\boldsymbol{\lambda}$ for which the distance between $\boldsymbol{x}$ and $\boldsymbol{F}(\boldsymbol{\lambda})$ is minimized,

$$\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{\lambda}} \|\boldsymbol{x} - \boldsymbol{F}(\boldsymbol{\lambda})\|_2 . \tag{1}$$

By definition, it is reasonable to assume that $\boldsymbol{F}$ is self-consistent, meaning that $\boldsymbol{F}(\boldsymbol{\lambda}) = E(\boldsymbol{x} | \boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}) = \boldsymbol{\lambda})$.

In fact, $\boldsymbol{F}$ defines a d-dimensional manifold that is embedded in a D-dimensional space. Meanwhile, $\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x})$ defines a special coordinate chart that projects point $\boldsymbol{x}$ onto the d-dimensional manifold. For a point $\boldsymbol{x}_0$ in $\mathcal{R}^D$, there is a low-dimensional representation provided by $\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}_0)$ and a corresponding projection point $\boldsymbol{F}(\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}_0))$. The line between $\boldsymbol{x}_0$ and $\boldsymbol{F}(\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}_0))$ is the projection direction that is orthogonal to the tangent plane because the manifold is locally homeomorphic with $\mathcal{R}^d$. Figure 2 illustrates the projection procedure.

With these notations, we are ready to define the adversarial region as follows.

Definition 1 (Adversarial Region): $\mathcal{D}$ is a dataset in D-dimensional space, and M is its manifold. For $\boldsymbol{x}_0 \in \mathcal{D}$, $\boldsymbol{x}_0^*$ is the projection point of $\boldsymbol{x}_0$ on M, i.e., $\boldsymbol{x}_0^* = \boldsymbol{F}(\boldsymbol{\lambda}_{\boldsymbol{F}}(\boldsymbol{x}_0))$. The adversarial region of $\boldsymbol{x}_0$ can be defined as

$$S_{AR}(\boldsymbol{x}_0) = \left\{ \boldsymbol{x} \middle| \boldsymbol{x} = \boldsymbol{x}_0 + \varepsilon \frac{\boldsymbol{x}_0 - \boldsymbol{x}_0^*}{\left\|\boldsymbol{x}_0 - \boldsymbol{x}_0^*\right\|_2}, \varepsilon \in [\varepsilon_{min}, \varepsilon_{max}] \right\}. \tag{2}$$

Here, ε_{min} guarantees that the adversarial examples deviate from the tangent plane far enough for misclassification, and ε_{max} guarantees that the distortion of $\boldsymbol{x}$ is imperceptible. In fact, the adversarial region defines a set whose elements

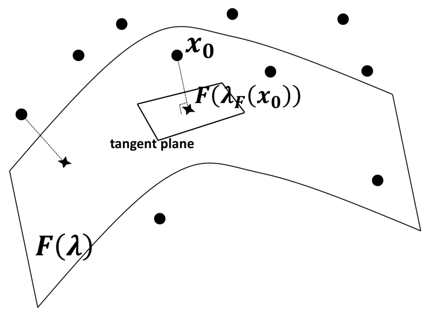


Fig. 2. A sample x_0 in D-dimensional space is projected onto the d-dimensional manifold via $F(\lambda_F(x_0))$.

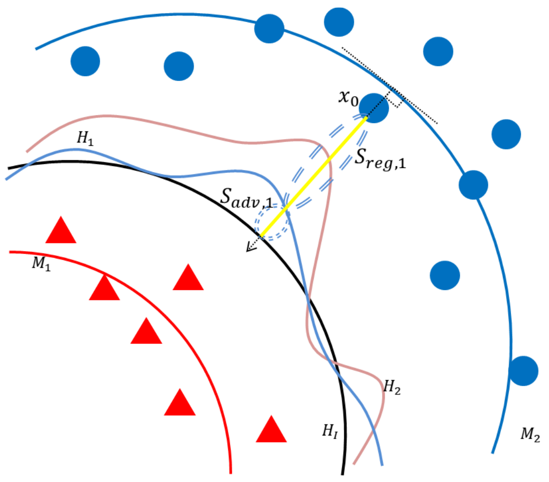


Fig. 3. There are two manifolds (M_1 and M_2) and some points sampled from these two manifolds. H_1 and H_2 are the decision boundaries of two different classifiers while H_I is the ideal decision boundary. The $S_{AR}(x_0)$ represented by the yellow solid line can be divided into four subsets (two regular subsets and two adversarial subsets) by H_1 and H_2. The regular subset ($S_{reg,1}$) and the adversarial subset ($S_{adv,1}$) obtained by H_1 are shown. Here, the intersection of two adversarial subsets S_I is actually $S_{adv,1}$.

move along the direction orthogonal to the tangent plane of the manifold.

Based on the definition of the adversarial region, it is clear that data points in adversarial regions impose a potential threat to all classifiers. Since different classifiers may have different decision hyperplanes, we can use these hyperplanes to divide the adversarial region into two subsets, the adversarial subset and regular subset. If there is an intersection (S_I) of two different adversarial subsets divided by two models, then both models will misclassify samples in S_I. In other words, the samples in the intersection of two adversarial subsets are able to transfer between two models. Figure 3 shows an example of S_I and S_{AR} in 2D space.

C. Principal Component Adversarial Example

To overcome the limitations of the model-dependent method, it is urgent to devise a target-free method. Because the adversarial region relies on a data manifold that is independent of the classification models, we can generate adversarial examples with an unsupervised method according to the definition of the adversarial region.

Suppose that $x \in \mathcal{R}^D$ is a training example from dataset $\mathcal{D}$ and that x^* is the projection point of x onto the manifold M, i.e., $x^* = F(\lambda_F(x))$. The adversarial examples of our method can be generated as follows:

$$x_{adv} = x + \varepsilon \frac{x - x^*}{\|x - x^*\|_2} \tag{3}$$

where ε controls the magnitude of the perturbation.

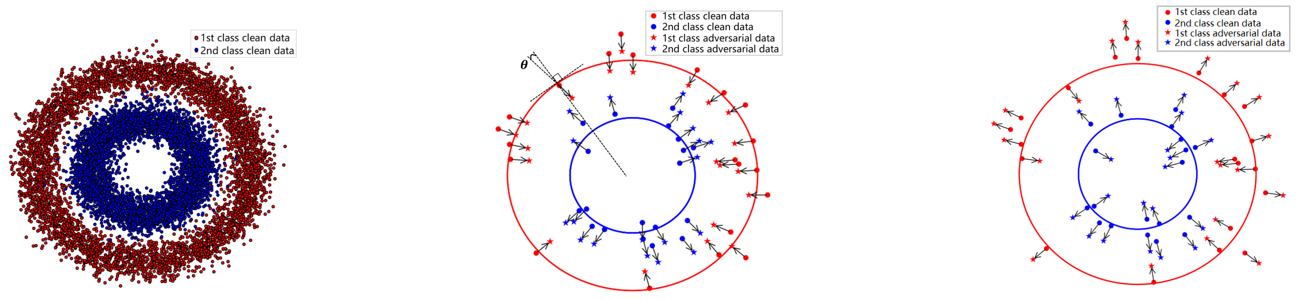
However, the manifold M is hard to explicitly construct, especially for complicated real-world datasets. Therefore, the projection $F(\lambda_F(x))$ cannot be calculated directly. In this paper, we approximate the manifold using principal component analysis (PCA) to generate adversarial examples.

Given a set of data points $\mathcal{D} = \{x_1, \ldots, x_n\}$, we apply PCA to obtain a transformation matrix $U \in \mathcal{R}^{D\times D}$ that consists of D principal components $\{u_1, u_2, \ldots, u_D\}$ with the corresponding eigenvalues $\{\gamma_1, \gamma_2, \ldots, \gamma_D\}$ in descending order. Specifically, we transform x to z, such that $z = U^T x$ and $z^* = U^T x^*$, where $z \in \mathcal{R}^D$ and $z^* \in \mathcal{R}^D$. We assume that the first k principal components can approximately construct the data manifold, i.e., $z_i = z_i^*$, for $i \in \{1, 2, \ldots, k\}$. Here, z_i and z_i^* refer to the i-th entries of vectors z and z^*, respectively. According to Eq. (3), adversarial examples of x can be generated as follows:

$$\begin{aligned} x_{adv} &= x + \frac{\varepsilon}{\|x - x^*\|_2}(x - x^*) \\ &= \sum_{i=1}^{k}\left[\left(1 + \frac{\varepsilon}{\|x - x^*\|_2}\right) z_i - \frac{\varepsilon}{\|x - x^*\|_2} z_i^*\right] u_i \\ &\quad + \sum_{i=k+1}^{D}\left[\left(1 + \frac{\varepsilon}{\|x - x^*\|_2}\right) z_i - \frac{\varepsilon}{\|x - x^*\|_2} z_i^*\right] u_i \\ &= \sum_{i=1}^{k} z_i u_i + \sum_{i=k+1}^{D}\left[z_i + \frac{\varepsilon z_i}{\|x - x^*\|_2} - \frac{\varepsilon z_i^*}{\|x - x^*\|_2}\right] u_i \\ &= x + \sum_{i=k+1}^{D} \frac{\varepsilon}{\|x - x^*\|_2}\left(\frac{z_i^*}{z_i} - 1\right) z_i u_i. \end{aligned} \tag{4}$$

Given x, $\|x - x^*\|_2$ is a constant, and ε is a parameter for controlling the noise amplitude. Here, we use $\beta(i) = \frac{z_i^*}{z_i} - 1$ to denote the weight of the small multiplicative distortion of component u_i, which changes according to the index i. We now show that $|\beta(i)|$ is an increasing function.

The eigenvalue γ_i corresponds to the variance of z_i, and the mean of z_i is 0. Since γ_i is sorted in descending order, z_i becomes closer to 0 as the index i increases from k to D. Because x^* is the projection point of x on the manifold, without loss of generality, z_i^* is a dependent variable of z_i, i.e., $z_i^* = g(z_i)$. Consequently, $g(z_i)$ can be approximated via a Maclaurin expansion of the second order, i.e., $g(z_i) \approx g(0) + g'(0) z_i + \frac{g''(0)}{2!} z_i^2$. Therefore, $|\beta(i)| = \left|\frac{g(0)}{z_i} + \frac{g''(0)}{2!} z_i + g'(0) - 1\right|$. It is then clear that $|\beta(i)|$ is an increasing function with respect to the index i. In other words, we can replace $|\beta(i)|$ with an increasing function, even though z^* is unknown. In our experiments, a linear increasing function works well, and the sign of the function has limited influence. Finally, we derive a method,



(a) Synthetic data

(b) Angles beteen $\nabla_{\boldsymbol{x}} J$ and manifold

(c) Perturbations generated by the manifold

Fig. 4. Synthetic data example. (a) Shows the synthetic dataset. (b) Illustrates the angle θ between $\nabla_{\boldsymbol{x}} J$ and the tangent directions, where J is the loss function of the neural network. The average of θ is 5.6°. (c) gives the adversarial examples generated by Eq. (3). The transfer rate of the adversarial examples in (c) is 83.9%, whereas the transfer rate of the adversarial examples generated by FGS is 80.3%.

Algorithm 1 Algorithm of Principal Components Adversarial Examples

```
Input: input data set $\mathcal{D}$, the component index $k$, the magnitude of perturbations $\epsilon$, and an increasing function $\beta(i)$
Output: principal component adversarial example set $\mathcal{D}_{adv}$
1: Apply PCA (or Kernel PCA) to $\boldsymbol{x}$:
   $\boldsymbol{z} = PCA\_transform(\boldsymbol{x})$
2: Scale $\boldsymbol{z}$ with $\beta(i)$:
   if $i > k$ : $\hat{z}_i = \beta(i) z_i$
   else : $\hat{z}_i = z_i$
3: Generate the perturbation $\boldsymbol{p}$:
   $\hat{\boldsymbol{x}} = PCA\_inverse(\hat{\boldsymbol{z}})$, $\boldsymbol{p} = \epsilon \frac{\hat{\boldsymbol{x}}-\boldsymbol{x}}{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|_2}$
4: Generate adversarial examples:
   $\boldsymbol{x}_{adv} = \boldsymbol{x} + \boldsymbol{p}$
5: return $\mathcal{D}_{adv}$;
```

called principal component adversarial examples, to generate adversarial examples, as shown in Algorithm 1.

In fact, we can simply transform $\boldsymbol{x}$ to $\boldsymbol{z}$, scale z_i with $\beta(i)$ for $i > k$, and take the inverse PCA transform to obtain the perturbed data. Therefore, this method can be readily extended to kernel PCA [36].

IV. Experiments

In this section, we conduct extensive experiments to demonstrate the proposed hypothesis and the effectiveness of our PCAE method. We verify the proposed adversarial region on one synthetic dataset and compare PCAE with FGS and CW on MNIST, CIFAR-10 and ImageNet. These datasets are widely used in the related papers [11], [20], [24], [37]. We select FGS and CW as our baseline because they have been shown to outperform other methods [20], [24]. From a security perspective, transferability is an important property of adversarial examples because it enables an attacker to create adversarial examples that could fool a target model. Hence, we focus on mainly the transferability in this paper, and we measure the transferability using the same criteria as in [24]. For candidate adversarial examples, we select those misclassified by the source model and then measure how many of them are misclassified by the other models. The misclassified fraction is the transfer rate. Finally, we show that PCAE is difficult to defend against even if the defender exactly understands the generation mechanism.

A. Verification of Adversarial Region

The manifold of real-world high-dimensional data points is difficult to explicitly compute. For simplicity of verifying the adversarial region, we generate one simple synthetic dataset containing data points distributed on two concentric circles, where each circle represents one class. This particular synthetic dataset has the advantage that the manifold of the data distribution is known. In fact, a similar synthetic dataset has been discussed in [25]. The distribution of the data is illustrated in Figure 4(a). There are 10000 points for each class. We use a fully connected neural network with one hidden layer to train a classifier using one-half of the data points. The classification accuracy is 99.4% when tested on the remaining data points.

As mentioned in our motivation, the sensitive directions of a classifier in terms of the input should be orthogonal to the data manifold. At point $\boldsymbol{x}$, the sensitive direction is therefore $\nabla_{\boldsymbol{x}} J$, where J is the loss function of the neural network. Figure 4(b) shows some randomly selected pairs of $\boldsymbol{x}$ and $\nabla_{\boldsymbol{x}} J$. As shown in Figure 4(b), we compute the average angle $\hat{\theta}$ between two vectors. One is the vector connecting the original data points and the center of the circle, and the other is the sensitive direction $\nabla_{\boldsymbol{x}} J$. The average of $\hat{\theta}$ is 5.6°, which is very close to the ideal angle 0°. This result validates our adversarial region hypothesis; that is, the most sensitive directions are those that are orthogonal to the manifold. Additionally, we generate adversarial examples according to Eq. (3). The transfer rate of these adversarial examples is 83.9%, while the transfer rate of FGS is 80.3%. These results validate the transferability of adversarial examples in the adversarial region. Hence, we speculate that neural networks are susceptible to the direction orthogonal to the manifold.

B. Comparison With Other Methods

We also conduct experiments on real datasets to validate the effectiveness of our PCAE generation method. Since there are no unsupervised method, we directly compare PCAE with two state-of-the-art supervised methods: FGS and CW. The results show that PCAE has better transferability across different models, especially when the data are insufficient or the target model is defensive.

FGS is a widely used adversarial example generation method [24] that estimates the imperceptible perturbation by taking the sign of the gradient $\hat{r}(\boldsymbol{x}) = \epsilon \operatorname{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\omega}, \boldsymbol{x}, y))$, where J is the cost used to train the neural network, $\boldsymbol{\omega}$ is the model parameters, and y is the label of $\boldsymbol{x}$. In the experiments, we reimplement the FGS method.

CW is the most transferable adversarial example creation method [20]. Instead of solving the original box-constrained optimization problem, CW uses an alternative formulation, $\min D(x, x+\delta) + c \cdot l(x+\delta)$, to construct adversarial examples, where D is a distance metric and l is an objective function such that the classifier gives an incorrect predication. For CW, we use the authors' open source implementations.

MagNet is an effective defensive framework that uses external neural network(s) to detect and reform an input image [38]. In the testing phase, MagNet rejects or reforms the input images according to the learned manifold of the clean images. Here, reforming the images means that the images are reformed to lie on the manifold. Then, the classifier is fed the reformed images. We use the same settings in [38] to defend the target model. Note that recent work [39] showed that CW can break MagNet, but the authors use a special objective function aimed at MagNet. In our experiments, we focus on mainly the original CW method, as we do not know which defense method is applied in practice.

Previous works [11], [40] indicate that more comprehensive notions of perceptual similarity are important. In this paper, we use the notation used in [11] for measuring the visual similarity between two images. Wang et al. noted that human visual perception is highly sensitive to the structural information of an image and proposed the structural similarity (SSIM) index as a measurement of image similarity [41]. Given a clean image $\boldsymbol{x}$ and its corresponding adversarial image $\boldsymbol{x}_{adv}$, SSIM$(\boldsymbol{x}, \boldsymbol{x}_{adv})$ measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{x}_{adv}$. A larger SSIM$(\boldsymbol{x}, \boldsymbol{x}_{adv})$ indicates a higher similarity between two images. The settings used in our experiments are the same as those in [11].

We train four neural networks with different architectures, $\{NN_0, NN_1, NN_2, NN_3\}$, to compare the transferability of FGS, CW and our PCAE. NN_0 is used to construct the FGS and CW adversarial examples. Our PCAE can generate adversarial examples directly, without neural networks or image labels. Following [24], adversarial examples misclassified by NN_0 are fed into NN_1, NN_2 and NN_3 to calculate the transfer rate. For the MNIST/CIFAR-10 dataset, we also decrease the number of training samples of NN_0 from 60K/50K to 6K/5K and 0.6K/0.5K to further explore the relationship between the transfer rate and the number of training samples.

In addition to the insufficient data case, we consider the case where the target model is defensive. We apply MagNet to the target models $\{NN_1, NN_2, NN_3\}$, and evaluate the transferability of the adversarial examples created by the different methods. We train MagNet with the training data and select a threshold using the validation set. More details can be found in [38].

TABLE I

TRANSFER RATE (%) OF THE ADVERSARIAL EXAMPLES GENERATED USING FGS, CW AND OUR PCAE ON MNIST

SSIM	Method	60K	6K	0.6K
0.70	FGS	29.02	11.05	4.31
	CW	**94.63**	**70.34**	**39.16**
	PCAE	72.87	53.16	32.30
0.75	FGS	30.78	10.02	2.92
	CW	**81.63**	**51.53**	18.30
	PCAE	64.86	42.78	**21.04**
0.80	FGS	32.42	7.69	2.30
	CW	53.2	16.43	6.28
	PCAE	**56.38**	**31.12**	**12.67**
0.85	FGS	28.51	6.14	1.69
	CW	17.74	3.56	2.00
	PCAE	**47.20**	**21.12**	**8.72**
0.90	FGS	24.58	4.53	1.01
	CW	2.55	-	-
	PCAE	**37.49**	**14.79**	**4.41**

1) Experiments on MNIST: MNIST contains 60,000 images for training and 10,000 for testing. Each sample is a gray image of size 28×28, comprising a hand-written digit. We adopt the LeNet-5 architecture as our source model NN_0. The remaining three network architectures are NN_1 (C20-relu-M2-C50-relu-M2-F500-relu), NN_2 (C32-relu-M2-C64-relu-M2-F500-relu-F200-relu), and NN_3 (C32-M2-C64-A2-F500-relu). C32, relu, M2, A2, F500, represent a convolution layer with 32 feature maps, the $relu$ activation function, $Max\text{-}pooling$ with a 2×2 kernel, $Average\text{-}pooling$ with a 2×2 kernel, and a $fully\text{-}connected$ layer with 500 nodes, respectively. The optimizer of each model is the same as that of LeNet-5. The accuracies of these models are greater than 99%.

Table I summarizes the results without defense. The adversarial examples created by PCAE have stronger transferability than those of CW and FGS when the similarity is high (SSIM $\geq$ 0.80). Moreover, the higher the similarity of the adversarial examples and the corresponding clean examples is, the stronger the transferability of PCAE. When more noise is allowed (SSIM $<$ 0.80), PCAE is still better than FGS, while CW can construct more transferable adversarial examples. However, if we reduce the number of training samples used by the source model (NN_0), the advantage of CW is weakened when SSIM is less than 0.80, and PCAE sometimes performs better than CW (SSIM $=$ 0.75 and training dat$a = 0.6k$). It is reasonable that the performance degrades when we decrease the number of training samples for NN_0, as the adversarial examples are selected from adversarial candidates using NN_0. There is an intriguing phenomenon that the adversarial examples generated by CW are so noisy that the SSIM value is always less than 0.90 when the training data are insufficient. It may be inferred that PCAE is better than FGS in most cases. Compared to CW, PCAE is a good choice for cases with high similarity. In addition, PCAE is much more robust against training samples than are CW and FGS. Figure 5 shows the adversarial examples generated using FGS, PCAE and CW on MNIST. Compared with FGS and

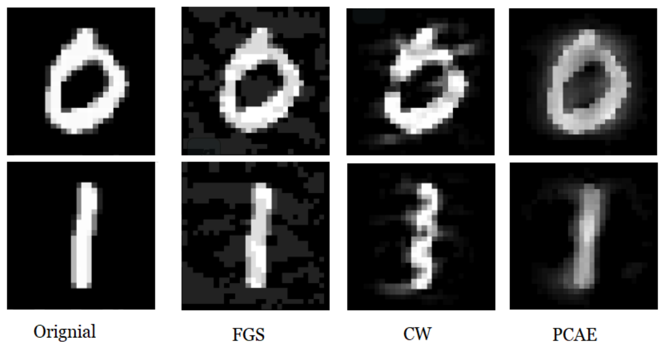


Fig. 5. The original images from MNIST and their adversarial examples generated by FGS, CW and PCAE, respectively. (SSIM=0.70).

CW, the perturbations in the adversarial examples generated via our PCAE are less imperceptible.

These results demonstrate the advantages of PCAE. Our method can construct stronger transferable adversarial examples than those created by some model-dependent methods. Moreover, under conditions of high similarity, the adversarial examples generated by our method are more transferable. Furthermore, PCAE has better performance than the model-dependent method when the data are insufficient.

In addition to inspiring us to devise an adversarial examples generation method, the notion of the adversarial region can also provide a clear understanding of the properties of target-free adversarial examples. The target-free method aims to find an adversarial region that is relevant to only the data distribution, which is unrelated to the decision boundary of the classifier. This processing is target-free, so adversarial examples that pose potential threats for any classifiers targeting the same classification task can be found. Thus, the target-free method is able to find more transferable adversarial examples. In addition, the target-free adversarial perturbations are perpendicular to the tangent plane of the data manifold, which means that the target-free adversarial examples have high similarity with points lying on the data manifold. Hence, the target-free adversarial examples have stronger transferability when the similarity is high. Furthermore, the model-dependent method focuses on finding adversarial perturbations that are orthogonal to the decision boundary of a specific classifier, which may lead to bad transferability when the classifier performs poorly. Therefore, the target-free method may perform better in cases with insufficient data.

Figure 8(a) further shows the transfer rate when the target model is defensive. "ori" and "ref" denote the transfer rate of the original adversarial examples and the reformed adversarial examples, respectively. It shows that PCAE outperforms both FGS and CW when the target model is defensive. Furthermore, PCAE is the most roust one since it has the smallest change in transfer rate before and after defense.

These results indicate that PCAE is not only able to bypass the detector but also maintains good transferability after the reformer. Thus, there is little performance degradation for PCAE. By contrast, CW and FGS can be blocked by MagNet, as reported in [38]. Therefore, the target-free method has an advantage over the model-dependent method in bypassing the defense method. MagNet has the ability to move adversarial examples towards the manifold [38]. The target-free method constructs adversarial examples using the manifold, which may result in target-free adversarial examples close to the manifold. Thus, the target-free adversarial examples, even the reformed examples, have good transferability. MagNet reduces the noise amplitude, so the target-free can bypass the defense.

TABLE II

Transfer Rate (%) of the Adversarial Examples Generated Using FGS, CW and Our PCAE on CIFAR-10

SSIM	Method	50K	5K	0.5K
0.70	FGS	90.92	78.44	45.69
	CW	**96.43**	**89.05**	52.1
	PCAE	77.34	60.84	**53.17**
0.75	FGS	89.18	72.16	38.48
	CW	**93.91**	**83.47**	45.12
	PCAE	71.88	53.01	**45.70**
0.80	FGS	86.40	65.07	31.35
	CW	**88.98**	**74.77**	37.11
	PCAE	62.86	43.58	**37.35**
0.85	FGS	80.86	56.56	23.65
	CW	**81.87**	**61.41**	26.37
	PCAE	54.52	34.42	**28.49**
0.90	FGS	**71.29**	**46.47**	17.25
	CW	70.76	45.4	19.22
	PCAE	42.99	24.22	**20.95**

2) Experiments on CIFAR-10: CIFAR-10 consists of 50,000 training and 10,000 testing images from 10 categories. Each image is an RGB image of size 32 × 32. The source model is the "cifar quick" architecture provided by caffe [42], and the architectures of tge other three models are NN_1 (C32-M3-relu-C64-relu-A3-C64-relu-A3-F64-relu), NN_2 (C32-M3-relu-C64-relu-A3-C64-relu-A3-F128-relu-Dropout, NN_3 (C32-M3-relu-C64-relu-M3-C64-relu-A3-F128-relu). We use the same optimizer for all the models. Here we use kernel PCA because the data distribution of CIFAR-10 is much more complicated than MNIST. The accuracies of these classifiers are approximately 77%, which is similar to the results reported in [20]. Note that except for subtracting the mean of the images, we do not perform any image augmentations.

Table II shows the transfer rates of FGS, PCAE and CW without defense. When the NN_0 is trained with 5K or 50K samples, the performances of FGS and CW are better than that of PCAE. However, PCAE is the most transferable method when we reduce the number of training samples. This result is expected because the manifold of CIFAR-10 is too difficult to approximate for kernel PCA. In addition, PCAE does not use any information from classifiers. However, the target-free method still has better performance than the model-dependent method when the data are insufficient, demonstrating that the proposed PCAE is more robust than FGS and CW.

From our adversarial region perspective, PCAE has poor performance because of the highly complex manifold. In contrast to neural networks, kernel PCA has limited power for approximating such difficult manifolds. Regardless, the target-free method demonstrates its robustness against the training samples.

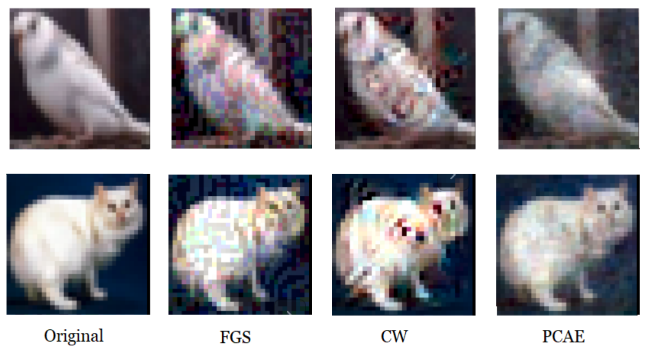


Fig. 6. The original images from CIFAR-10 and their adversarial examples generated by FGS, CW and PCAE, respectively. (SSIM=0.70).

The results of the defensive setting of the target model are shown in Figure 8(b). First, the black solid line is always at the bottom because MagNet defends against the FGS attack completely. Second, the gap between the red solid line and the red dotted line is the smallest; that is, the performance of PCAE suffers limited degradation. Third, the red solid line is above the blue solid line when SSIM < 0.80, and the situation is reversed when SSIM > 0.80, which means that PCAE has stronger transferability than CW when SSIM is low.

Figure 6 shows the adversarial examples generated using FGS, PCAE and CW when SSIM=0.70. The noise added to the clean images by PCAE is less imperceptible than that added by FGS and CW.

3) Experiments on ImageNet: In addition to considering MNIST and CIFAR-10, which are both relatively low-dimensional datasets, we conduct experiments on the ImageNet dataset. As a challenging classification dataset, ImageNet contains millions of high-resolution images [43], and convolutional neural network models pretrained on ImageNet have been widely adopted to many tasks [44], [45]. Following [20], we use the first 1000 images correctly classified by the source model. In contrast to the previous subsection, we use CaffeNet [42] and GoogLeNet [3] as our target models, which achieve 57.4% and 68.7% top-1 accuracy. For the model-dependent methods, we take advantage of AlexNet [1], whose top-1 accuracy is 57.1%, as the source model.

Table III shows that the CW method has a higher transfer rate than PCAE and FGS, regardless of whether the target model has a similar (CaffeNet) or different (GoogLeNet) architecture. This is because for each image CW finds adversarial perturbations by complex optimization which is time consuming. As the best model-dependent method, CW has stronger transferability than FGS and PCAE on the ImageNet dataset. When the target model is CaffeNet, FGS performs better than PCAE. However, when the target model is GoogLeNet, the performance of PCAE is much better than that of FGS. FGS is better than PCAE for transferring between similar network architectures, but when the target model and the source model have substantial differences, PCAE is better.

The adversarial region perspective can provide a clear understanding of these results. The CW method has the ability to accurately fit the decision boundary, as it finds adversarial perturbations by optimization. Without loss of generality, the fit of the data distribution determines the strength of the generalization ability. Thus, if the source model has high generalization ability, the model may fit the data distribution well. Consequently, the CW perturbations and the manifold are close to vertical; that is, the perturbations found by the CW method are in the set of the adversarial region when the source model performs well. Hence, the CW method has the strongest transferability on the ImageNet dataset. In contrast to the CW method, FGS has limited ability to fit the decision boundary, which can lead to two problems. First, it may cause FGS to have worse transferability than CW, even through the source model and the target model have similar architectures and performance. However, classifiers that have similar architecture may have similar decision boundaries [25], so FGS can take advantage of more information from the target model than can PCAE. Second, it may lead to a poor fit to the data distribution. Consequently, FGS suffers considerable performance degradation when the target model and source model have different architectures. Thus, we can understand why FGS performs better than PCAE on CaffeNet and why FGS performs worse than PCAE on GoogLeNet. Figure 7 shows that the adversarial perturbations generated by PCAE are less imperceptible than that generated by FGS and CW.

TABLE III

TRANSFER RATE (%) OF THE ADVERSARIAL EXAMPLES GENERATED USING FGS, CW AND OUR PCAE ON IMAGENET. OUR TARGET-FREE (UNSUPERVISED) METHOD ACHIEVES COMPARABLE PERFORMANCE WHEN COMPARED WITH MODEL-DEPENDENT (SUPERVISED) FGS METHOD. CW, ALSO A SUPERVISED METHOD, PERFORMS THE BEST BECAUSE IT FINDS ADVERSARIAL PERTURBATIONS BY COMPLEX OPTIMIZATION WHICH IS TIME CONSUMING

SSIM	CaffeNet			GoogLeNet		
	CW	FGS	PCAE	CW	FGS	PCAE
0.70	99.4	84.1	82.6	78.9	53.8	58.2
0.75	98.8	79.7	74.6	72.3	45.5	52.8
0.80	97.0	74.4	74.0	61.5	37.3	42.6
0.85	94.0	68.0	67.4	52.2	29.7	37.0
0.90	90.5	59.9	56.3	42.7	21.8	25.8

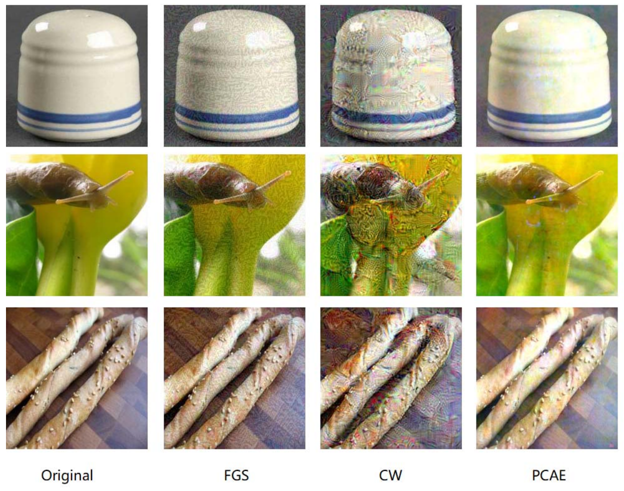
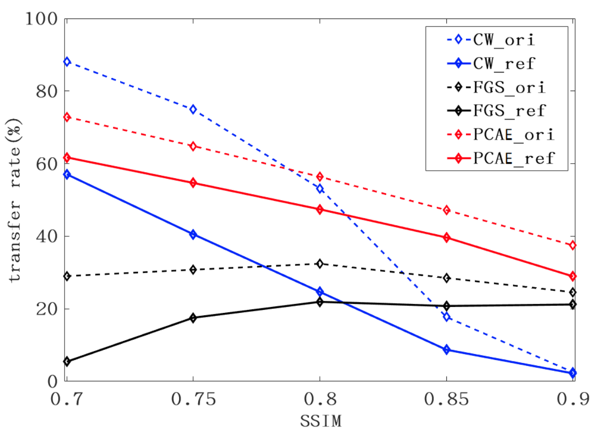
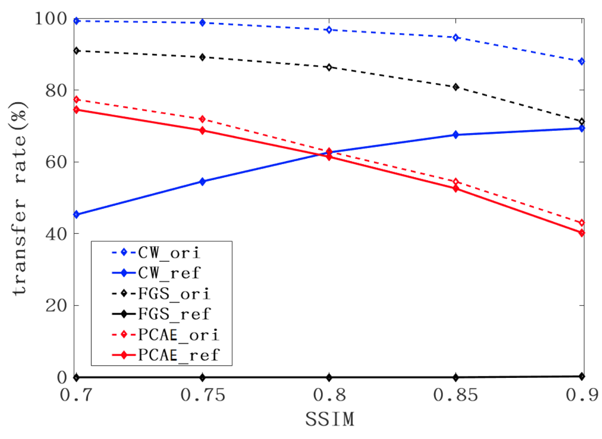


Fig. 7. The original images from ImageNet and their adversarial examples generated by FGS, CW and PCAE, respectively. (SSIM=0.70).



(a) MNIST



(b) CIFAR-10

Fig. 8. Changes in transfer rate before and after the defense of MagNet. We use the dotted line to indicate the transfer rate of the original adversarial examples and the solid line to indicate the transfer rate of the reformed adversarial examples.

4) The Attack Rate of PCAE: Target-free methods create adversarial examples using the data manifold instead of the source model, so we cannot calculate the attack rate of target-free methods. In fact, we usually select some of the adversarial examples misclassified by a model (NN_0) from all adversarial examples constructed by the target-free method. Thus, we can regard NN_0 as the source model and calculate the attack rate. We use "X-N" to denote the source model that is trained on the dataset X with N training samples. For ImageNet dataset, we utilize AlexNet as our source model. Note that the source model is used for the selection instead of constructing adversarial examples.

Table IV presents the attack rate results of PCAE on seven source models. The attack rate is somewhat low but reasonable. First, in contrast to the model-dependent method, the model-free method does not take advantage of any information of the source model. Thus, the adversarial examples generated for the decision surface are more specific than the adversarial examples generated for the manifold. Second, we focus mainly on the existence and the properties of the model-free adversarial examples, so we utilize a simple method to estimate manifolds. Note that the attack rate of PCAE on MNIST is low, which may be due to the dataset since the images in the MNIST dataset are so simple that even a small amount of noise will cause large virtual differences. We verify this speculation using FGS and find that the attack rates of both methods are comparable. The attack rates of FGS on MNIST are given in Table V. We can see that the attack rates of PCAE (unsupervised) are comparable to FGS (supervised method) on MNIST, which is consistent with our speculation.

TABLE IV

ATTACK RATE (%) OF PCAE ON DIFFERENT SOURCE MODEL. THE SUBSCRIPTS ARE USED TO REPRESENT THE NUMBER OF ADVERSARIAL EXAMPLES GENERATED WITH ONE INPUT

SSIM	0.90	0.85	0.80	0.75	0.70
MNIST-60k	1.30	2.45	4.27	8.88	17.09
MNIST-6k	3.72	5.43	7.45	12.04	19.35
MNIST-0.6k	12.10	14.85	17.64	22.78	29.66
CIFAR-50k	16.66	21.09	26.28	33.57	40.65
CIFAR-5k	34.70	35.57	37.63	39.48	39.68
CIFAR-0.5k	62.64	63.51	64.65	65.03	65.15
AlexNet	16.50	24.48	33.08	43.07	54.44

TABLE V

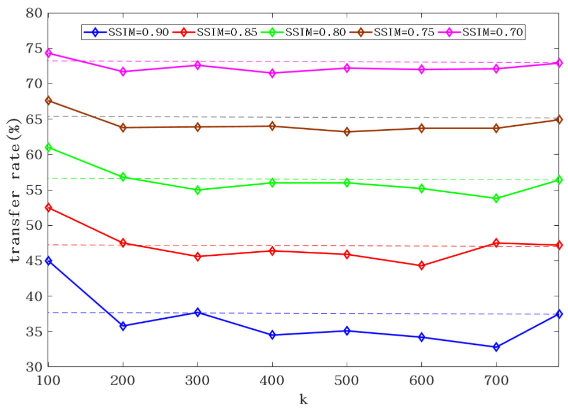
ATTACK RATE (%) OF FGS ON MNIST

SSIM	0.90	0.85	0.80	0.75	0.70
MNIST-60k	1.36	2.45	4.01	8.61	24.95
MNIST-6k	4.00	6.01	8.56	13.91	30.67
MNIST-0.6k	13.34	16.81	20.91	29.07	40.65

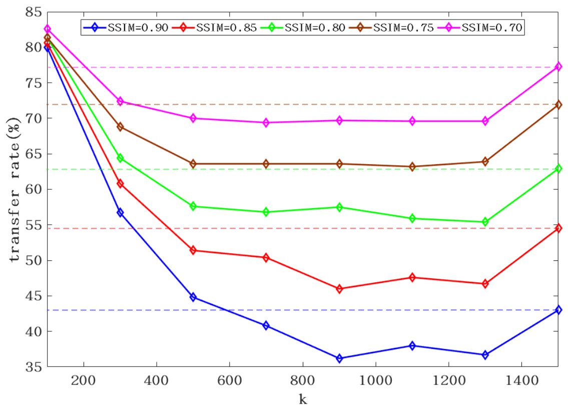
C. Defense Method for PCAE

We can devise a new loss function to defeat a specific defender [39]. Therefore, the defender may construct a method for a specific attack. To demonstrate the effectiveness of PCAE, we design a defensive method against PCAE. Before being fed to the target model, these adversarial examples are preprocessed by the defense method. We conduct experiments on the MNIST and CIFAR-10 datasets with the same settings as in the previous section. Surprisingly, PCAE suffers limited performance degradation, even though the defender knows the generation mechanism of PCAE.

In this paper, we approximate the manifold using PCA. Consequently, we can scale each coefficient (z_i) with $\beta(i)$ to generate adversarial examples according to Formula 4. Therefore, utilizing $\frac{1}{\beta_i}$ to rescale z_i is an intuitive defense against PCAE. However, several difficulties make this process impossible. First, the defender may apply a different transform matrix to PCAE, which may lead to different transform domain coefficients since different samples may derive different transform matrices. Second, the parameter $\beta(i)$ is untouchable for the defender. Third, we often inject noise into $\beta(i)$, which makes it impossible to reconstruct the clean samples. Eliminating low-energy components is another intuitive denoising method [46]. This strategy assumes that the first k components



(a) MNIST



(b) CIFAR-10

Fig. 9. Changes in transfer rate before and after the specific defense of PCAE. We use the dashed lines to indicate the transfer rate before PCA processing. k represents the number of components that are retained.

are uncontaminated and eliminates the noisy components. In this paper, we apply the second method to defend PCAE.

Figure 9 illustrates the difference in transfer rates before and after being defended. Solid lines of the same color are above the dotted lines when k is small, which means that the transfer rate is higher than the original transfer rate. When k is close to the data dimension, the solid lines are close to the dotted lines; that is, there is limited transfer rate degradation if few components are eliminated. This result is reasonable because k represents the reserved information. Substantial amounts of dropped information result in poor image quality, which may result in reduced model accuracy. When we drop minimal information, more adversarial information is reserved, which results in stronger adversarial properties. k can be adjusted to shift the solid line below the dotted line. Thus, the defender can find a k to reduce the transferability of PCAE.

We can conclude from these results that it is possible to devise a defense method to reduce the transferability of PCAE. However, the defensive parameter (k) is important, and even an appropriate parameter has limited influence on the transferability of PCAE. Hence, it is difficult to defend against PCAE, even though the defender know the generation mechanism.

D. The Influence of the $\beta(\cdot)$ Function

$\beta(i)$ is used for weighting the component z_i that is obtained by PCA (Sec. III.C.). To provide a full study, we exploit the influence of $\beta(i)$ on the performance of PCAE. The preceding discussion theoretically shows that the weight function should be an increasing function. To verify the conclusion, we evaluate the transfer rate of both increasing and decreasing functions on MNIST. In addition, we also consider a variety of functions, including linear, logarithmic and quadratic functions.

The considered functions are in the form of $\beta(i) = ai + b$, $\beta(i) = \ln(ai + b)$ and $\beta(i) = (ai + b)^2$ for linear, logarithmic and quadratic functions, respectively. All parameters are determined by grid search. Results are given in Figure 10. The difference between solid (increasing functions) and dashed lines (decreasing functions) shows that all decreasing functions substantially degrade the performance, which demonstrates

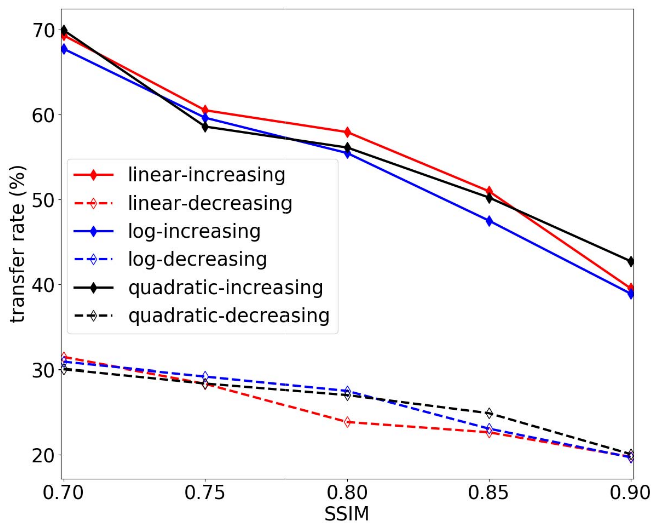


Fig. 10. Transfer rates when different functions are used for $\beta(\cdot)$.

that $\beta(i)$ should be an increasing function. It also shows that all increasing functions can achieve good transfer rate, which indicates that PCAE is robust to the function selection.

V. Conclusion and Future Work

In this paper, we consider the question of whether it is possible to construct adversarial examples without a classifier. This process is completely different from the current black-box or white-box attacks, as all methods in the literature use a classifier to create adversarial examples. To address this question, we consider the properties of adversarial examples and propose a concept, called the adversarial region, to understand the adversarial example phenomenon. The notion of the adversarial region provides an explanation of the existence of adversarial examples and makes it possible to construct adversarial examples without a classifier. On the basis of the adversarial region, we devise a target-free adversarial example generation algorithm using PCA. Remarkably, extensive experiments on both synthetic and real datasets verify that the adversarial examples generated by our proposed method have competitive transferability, especially when the target model is defensive or the training data are insufficient. However, we apply PCA to approximate the manifold in our work, which may reduce the performance of the target-free method. Thus, we will utilize more powerful manifold estimation methods to generate adversarial examples in the future.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[3] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[4] J. Li, H. Chang, J. Yang, W. Luo, and Y. Fu, "Visual representation and classification by learning group sparse deep stacking network," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 464–476, Jan. 2018.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[6] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[7] W. Wang, J. Shen, and L. Shao, "Video salient object detection via fully convolutional networks," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 38–49, Jan. 2018.

[8] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," 2016, *arXiv:1605.07277*. [Online]. Available: http://arxiv.org/abs/1605.07277

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*. [Online]. Available: http://arxiv.org/abs/1412.6572

[10] S. Sabour, Y. Cao, F. Faghri, and D. J. Fleet, "Adversarial manipulation of deep representations," 2015, *arXiv:1511.05122*. [Online]. Available: http://arxiv.org/abs/1511.05122

[11] A. Rozsa, E. M. Rudd, and T. E. Boult, "Adversarial diversity and hard positive generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2016, pp. 25–32.

[12] P. Tabacof and E. Valle, "Exploring the space of adversarial images," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 426–433.

[13] A. Fawzi, O. Fawzi, and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," 2015, *arXiv:1502.02590*. [Online]. Available: http://arxiv.org/abs/1502.02590

[14] C. Lyu, K. Huang, and H.-N. Liang, "A unified gradient regularization family for adversarial examples," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2015, pp. 301–309.

[15] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," 2014, *arXiv:1412.5068*. [Online]. Available: http://arxiv.org/abs/1412.5068

[16] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 427–436.

[17] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "Robustness of classifiers: From adversarial to random noise," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1632–1640.

[18] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, and S. Soatto, "Analysis of universal adversarial perturbations," 2017, *arXiv:1705.09554*. [Online]. Available: http://arxiv.org/abs/1705.09554

[19] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[20] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.

[21] A. Rozsa, M. Gunther, and T. E. Boult, "Are accuracy and robustness correlated," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2016, pp. 227–232.

[22] C. Szegedy *et al.*, "Intriguing properties of neural networks," 2013, *arXiv:1312.6199*. [Online]. Available: http://arxiv.org/abs/1312.6199

[23] T. Tanay and L. Griffin, "A boundary tilting persepective on the phenomenon of adversarial examples," 2016, *arXiv:1608.07690*. [Online]. Available: http://arxiv.org/abs/1608.07690

[24] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," 2016, *arXiv:1611.01236*. [Online]. Available: http://arxiv.org/abs/1611.01236

[25] J. Gilmer *et al.*, "Adversarial spheres," 2018, *arXiv:1801.02774*. [Online]. Available: http://arxiv.org/abs/1801.02774

[26] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "PixelDefend: Leveraging generative models to understand and defend against adversarial examples," 2017, *arXiv:1710.10766*. [Online]. Available: http://arxiv.org/abs/1710.10766

[27] H. Lee, S. Han, and J. Lee, "Generative adversarial trainer: Defense to adversarial perturbations with GAN," 2017, *arXiv:1705.03387*. [Online]. Available: http://arxiv.org/abs/1705.03387

[28] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "The robustness of deep networks: A geometrical perspective," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 50–62, Nov. 2017.

[29] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 86–94.

[30] P. Simard, B. Victorri, Y. LeCun, and J. Denker, "Tangent prop-a formalism for specifying selected invariances in an adaptive network," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 1992, pp. 895–903.

[31] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[32] S. Rifai, Y. N. Dauphin, P. Vincent, Y. Bengio, and X. Müller, "The manifold tangent classifier," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2011, pp. 2294–2302.

[33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[34] P. Vincent and Y. Bengio, "Manifold parzen windows," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2003, pp. 849–856.

[35] L. Cayton, "Algorithms for manifold learning," Univ. Ca San Diego, San Diego, CA, USA, Tech. Rep. CS2008-0923, 2005, pp. 1–17, vol. 12.

[36] G. H. Bakır, A. Zien, and K. Tsuda, "Learning to find graph preimages," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2004, pp. 449–456.

[37] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," 2017, *arXiv:1702.06280*. [Online]. Available: http://arxiv.org/abs/1702.06280

[38] D. Meng and H. Chen, "MagNet: A two-pronged defense against adversarial examples," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2017, pp. 135–147.

[39] N. Carlini and D. Wagner, "MagNet and 'Efficient defenses against adversarial Attacks' are not robust to adversarial examples," 2017, *arXiv:1711.08478*. [Online]. Available: http://arxiv.org/abs/1711.08478

[40] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv:1706.06083*. [Online]. Available: http://arxiv.org/abs/1706.06083

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[42] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Int. Conf. Multimedia (MM)*, 2014, pp. 675–678.

[43] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Apr. 2015.

[44] X.-S. Wei, J.-H. Luo, J. Wu, and Z.-H. Zhou, "Selective convolutional descriptor aggregation for fine-grained image retrieval," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2868–2881, Jun. 2017.

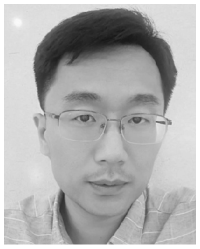
[45] T. Sun, Y. Wang, J. Yang, and X. Hu, "Convolution neural networks with two pathways for image style recognition," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4102–4113, Sep. 2017.

[46] M. Lebrun, "An analysis and implementation of the BM3D image denoising method," *Image Process. Line*, vol. 2, pp. 175–213, Aug. 2012.

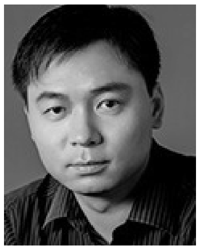
Yonggang Zhang received the B.S. degree from the Department of Information and Control Engineering, China University of Petroleum (East China), Qingdao, China, in 2017. He is currently pursuing the Ph.D. degree with the CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, University of Science and Technology of China, Hefei, China. His research interests include computer vision and machine learning.

Xinmei Tian (Member, IEEE) received the B.E. and Ph.D. degrees from the University of Science and Technology of China in 2005 and 2010, respectively. She is currently an Associate Professor with the CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China. Her current research interests include multimedia information retrieval and machine learning. She received the Excellent Doctoral Dissertation of Chinese Academy of Sciences Award in 2012 and the Nomination of National Excellent Doctoral Dissertation Award in 2013.

Ya Li received the B.S. and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), in 2013 and 2018, respectively. His research interests include machine learning and computer vision.



Xinchao Wang received the degree (Hons.) from The Hong Kong Polytechnic University (HKPU) in 2010 and the Ph.D. degree from the Ecole Polytechnique Federale de Lausanne (EPFL) in 2015. He was an SNSF Postdoctoral Fellow with the University of Illinois Urbana–Champaign (UIUC). He is currently a tenure-track Assistant Professor with the Stevens Institute of Technology, New Jersey, USA. His articles have been published in major venues, including CVPR, ICCV, ECCV, NeurIPS, AAAI, IJCAI, MICCAI, TPAMI, IJCV, TIP, TMI, and TNNLS. His research interests include artificial intelligence, computer vision, machine learning, medical image analysis, and multimedia. He serves as an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCSVT) and *Journal of Visual Communication and Image Representation* (JVCI), as a Senior Program Committee member of AAAI 2019 and IJCAI 2019 and 2020, and as an Area Chair of ICME 2019 and ICIP 2019.



Dacheng Tao (Fellow, IEEE) is currently a Professor of computer science and an ARC Laureate Fellow with the School of Computer Science and the Faculty of Engineering, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Centre, The University of Sydney. His research results in artificial intelligence have expounded in one monograph and more than 200 publications at prestigious journals and prominent conferences, such as IEEE T-PAMI, IJCV, JMLR, AIJ, AAAI, IJCAI, NIPS, ICML, CVPR, ICCV, ECCV, ICDM, and KDD, with several best paper awards. He is a Fellow of AAAS, ACM, and Australian Academy of Science. He received the 2018 IEEE ICDM Research Contributions Award and the 2015 Australian Scopus-Eureka Prize.