Detecting Adversarial Examples on Deep Neural Networks with Mutual Information Neural Estimation

Song Gao, Ruxin Wang, Xiaoxuan Wang, Shui Yu, *Fellow, IEEE*, Yunyun Dong, Wei Zhou, *Member, IEEE*, Shaowen Yao, *Member, IEEE*

Abstract—Despite achieving exceptional performance, deep neural networks (DNNs) suffer from the harassment caused by adversarial examples, which are produced by corrupting clean examples with tiny perturbations. Many powerful defense methods have been presented such as training data augmentation and input reconstruction which, however, usually rely on the prior knowledge of the targeted models or attacks. A clean example and its adversarial version are very similar but have different high-level representations in a victim model. If we can obtain a space in which the representations of similar examples are also similar, then adversarial examples can be picked out by comparing the representations of input examples in this space and the high-level space of the victim model. Inspired by this, we propose a novel approach for detecting adversarial images, which can protect any pre-trained DNN classifiers and resist an endless stream of new attacks. Specifically, we first adopt a dual autoencoder to project images to a latent space. The dual autoencoder uses the self-supervised learning to ensure that small modifications to samples do not significantly alter their latent representations. Next, the mutual information neural estimation is utilized to enhance the discrimination of the latent representations. We then leverage the prior distribution matching to regularize the latent representations. To easily compare the representations of examples in the two spaces, and not rely on the prior knowledge of the targeted model, a simple fully connected neural network is used to embed the learned representations into an eigenspace, which is consistent with the output eigenspace of the targeted model. Through the distribution similarity of an input example in the two eigenspaces, we can judge whether the input example is adversarial or not. Extensive experiments on MNIST, CIFAR-10, and ImageNet show that the proposed method has superior defense performance and transferability than state-of-the-arts.

Index Terms—Adversarial examples, detection, mutual information neural estimation, prior distribution matching.

1 Introduction

As a powerful tool in artificial intelligence, deep neural networks (DNNs) have been widely applied in many perceptual tasks, such as image classification, semantic segmentation, and speech recognition. Especially in image classification, the performance of deep learning-based classifiers even surpasses that of humans. As DNNs are becoming ever more prevalent, the concerns on the security of deep models are simultaneously raised. Several researchers [1], [2] have demonstrated that DNNs are sensitive to adversarial attacks, that is, the elaborately designed subtle perturbations in original examples can mislead DNN models to generate wrong results [3], [4], [5], [6]. A practical concern is that the perturbations are imperceptible to human eyes, but can fool DNNs with high confidence. Hence, this undesirable characteristic enhances the challenge of DNNs in safety-critical applications.

- *S. Gao, Y. Dong, W. Zhou and S. Yao are with the Engineering Research Center of Cyberspace and the National Pilot School of Software, Yunnan University, Kunming 650504, China, (e-mail: {gaos, dongyy929, zwei, yaosw}@ynu.edu.cn);*
- *R. Wang is with the Alibaba Group, Beijing 100026, China, (e-mail: rosinwang@gmail.com);*
- *X. Wang is with the School of Information Science and Technology, Yunnan Normal University, Kunming 650504, China, (e-mail: wangxiaoxuan1037@163.com);*
- *S. Yu is with the School of Computer Science, University of Technology Sydney, Sydney 2007, Australia, (e-mail: Shui.Yu@uts.edu.au)*

(Corresponding authors: Wei Zhou and Shaowen Yao.)

Various defense methods have been proposed to attempt to remedy the issues of adversarial examples. Some of these methods generally focus on the solutions in the training phase, such as distilling the targeted model [7], [8], adding regularization to the cost function [9], [10], and augmenting the training data [11], [12]. However, when a well-trained model is applied, the cost of retraining to cope with new attacks is enormous, especially in the case that there are always new powerful adversaries that can attack successfully. On the other hand, some trials focus on removing adversarial perturbations before feeding input examples to the targeted model [13], [14], [15], [16]. But preprocessing input examples could lead to the loss of prediction accuracy.

Considering these difficulties, the detection-based defense strategies have attracted a lot of attention recently as optional solutions. Lu et al. [17] utilized a RBF-SVM classifier with discrete codes generated from high-level ReLUs to detect adversarial samples. Metzen et al. [18] trained an auxiliary network which uses the outputs of the middle-layers as features to predict the probability of an input being adversarial. [19], [20], and [21] leverage the distribution characteristics of different categories at the hidden layers of the deep model to distinguish adversarial examples. These defense methods depend closely on the prior knowledge of the targeted model, hence being model-specific, in which case the robustness improvement of a model cannot be transferred to other models. [22] and [23] train DNN-based

binary classifiers as detectors to identify adversarial examples. These methods can get rid of the dependence on the targeted model, but still require the assistance of attacks. They are unstable and prone to fail in resisting stronger attacks.

In this work, we propose a novel approach for detecting adversarial images. Firstly, a dual autoencoder [24], which is harnessed to impose the reconstruction constraint on the latent representations and their noisy versions, is adopted to project images into a latent space. Normally, adversarial perturbations will be progressively amplified by a victim model and lead to incorrect results, which means the high-level representations of an image and its adversarial version in the victim model are different. The self-supervised learning of the dual autoencoder guarantees that the latent representations of an image and its adversarial version are similar. We then utilize the global and local mutual information (MI) estimation [25], [26] for representation learning, by maximizing the global and local MI between the inputs and the outputs of the encoder to enhance the discriminability of the learned latent representations. In addition, we combine the mutual information maximization with the prior distribution matching in a way similar to the adversarial autoencoder (AAE) [27] to regularize the learned latent representations. After the dual autoencoder training, we only keep the encoder as a converter from the original space to the latent space. At this point, the only thing we can guarantee is that a clean image and its adversarial version are similar in the latent space, but different in the high-level space of the targeted model. In order not to depend on the prior knowledge of the targeted model, we choose the output eigenspace of the targeted model as the high-level space. Also, to easily compare the representations of images in different eigenspaces, we use a simple fully connected neural network to project the learned latent representations to an eigenspace, which is consistent with the output eigenspace of the targeted model. By comparing the distributions of an input image in the two output eigenspaces, we can judge whether it is adversarial or not. It is worth noting that our method regards the targeted model as a black box where only is the model output information used. Therefore, our method is model-agnostic, meaning that it has good transferability and can be reused to protect different models after training. Meanwhile, the proposed method does not use any adversarial examples in the training process. It has good generalization, as long as an adversarial example adheres to the principle that it is similar to its clean version but can fool the targeted model, our method can effectively capture it.

In summary, this work makes the following contributions:

- We propose a novel defense approach for detecting adversarial examples. The proposed approach does not depend on the details of the targeted model and thus exhibits good transferability among different models. Notably, our method does not rely on any prior knowledge of attacks and hence, it has good generalization and can defend against the endless stream of attacks.
- We present a joint learning framework to obtain good representations of the input images. This framework adopts a dual autoencoder architecture to improve the robustness of the learned representations on noise, utilizes the mutual information maximization to enhance the discriminability of the learned representations, and leverages prior distribution matching to regularize the learned representations.
- Extensive experiments on three real datasets verify that our approach achieves the state-of-the-art performance on resisting adversarial examples.

The remainder of this paper is organized as follows: We discuss the related work in Section 2. Section 3 presents the proposed approach in details. The experimental settings, results, and correlation analyses are shown in Section 4. Finally, Section 5 draws the conclusions.

2 Related Work

In this section, we briefly review the existing researches on the mutual information estimation, adversarial attacks and defense methods, which are closely related to this study.

2.1 Mutual Information Estimation

Mutual information is a measure based on the Shannon entropy of dependence between random variables. The MI between X and Y can be considered as the diminution of the nondeterminacy in X given Y:

$$I(X,Y) = H(X) - H(X|Y), \tag{1}$$

where H denotes the Shannon entropy, and $H(X|Y)$ denotes the conditional entropy of X given Y. MI has always been difficult to calculate. Accurate calculation of MI is only applicable to discrete variables, or to limited tasks with known probability distributions. For more general tasks, estimating MI could be accomplished by a feasible alternative solution. Many approaches to estimate MI have been proposed from adopting non-parametric kernel-density estimators [30], binning [31], and Parzen window [32] to utilizing Edgeworth expansion [33] and likelihood-ratio estimators [34]. Unfortunately, these methods are often hard to adapt for deep neural networks. Mutual Information Neural Estimation (MINE) [25] utilizes adversarial learning to estimate the MI of continuous variables, which makes it possible to compute MI between high dimensional input/output pairs of DNNs. Deep InfoMax (DIM) [26] extends MINE to learn useful representations. We are inspired by DIM to learn the discriminative representations of images in this work. In this way, the distances between different images will be increased in the latent space and correspondingly, the relative distances between similar images (e.g., a clean image and its adversarial version) will be decreased.

2.2 Crafting Adversarial Examples

Let $x \subset R^m$ be a legitimate image, and y be the corresponding class label. For a well-trained DNN model f with the parameters θ, $f(x,\theta) = y$. Adversarial attack is to find the adversarial example by

$$min\ ||r||\ \ subject\ to\ \ f(x+r,\theta) \neq y, \tag{2}$$

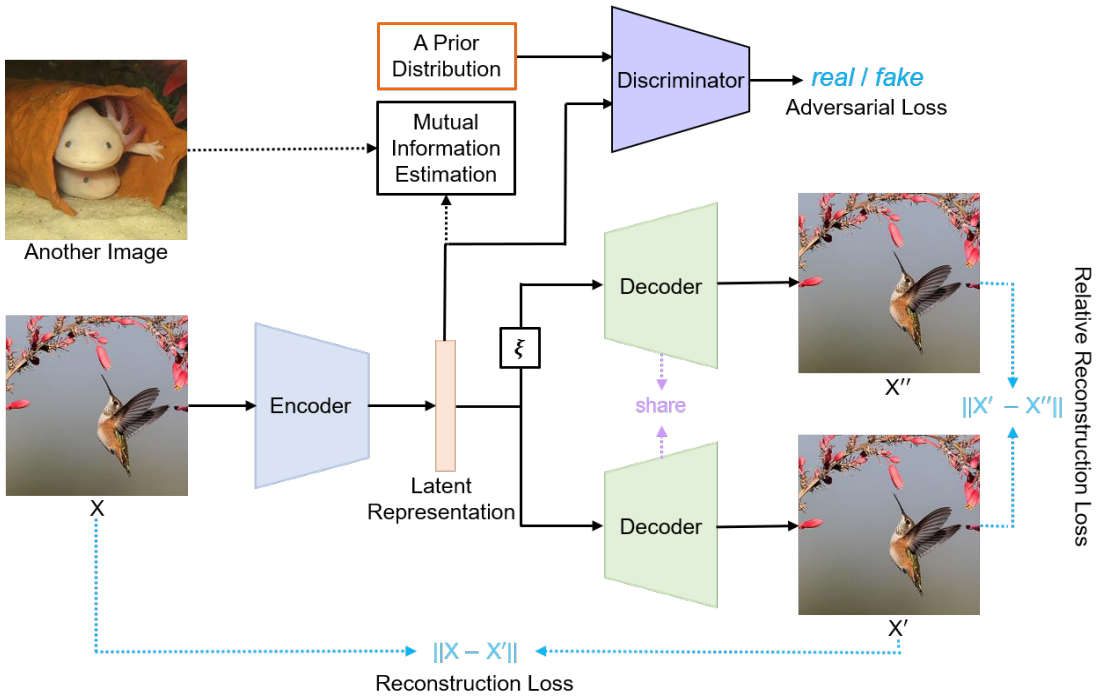


Fig. 1. Illustration of the latent representation acquisition framework. The encoder encodes the input images into a latent space, and the decoder reconstructs the inputs by the latent representations and their noisy versions. Meanwhile, the mutual information estimation including global MI estimation and local MI estimation, which will be clear later, is leveraged to improve the discriminability of the latent representations. In addition, a discriminator is trained adversarially to match the latent representations to a prior distribution.

where r denotes the adversarial perturbation, and $x + r$ is the adversarial example x_{adv}, i.e., $x_{adv} = x + r$. Since Szegedy et al. [1] first noticed the existence of adversarial examples, many attack methods have been presented to craft the worst-case perturbations. Here, we briefly introduce several adversarial attacks which are evaluated against the detection task in this work.

Fast Gradient Sign Method (FGSM) [2] is one of the earliest strategies for designing adversarial perturbations, which performs one-step update along the direction of the gradient at the current model state. The formula for FGSM is:

$$x_{adv} = x + \varepsilon \cdot sign\left(\nabla_x L\left(f\left(x, \theta\right), y\right)\right), \tag{3}$$

where ε is a constant that controls the maximal change of each pixel. $sign$ denotes the symbolic function, and $L(\cdot)$ denotes the loss function. ∇_x represents the gradient of model f with respect to the input x, and y is the ground-truth label of x.

FGSM is a simple and effective method to craft adversarial examples. Kurakin et al. [3] extended FGSM to an iterative algorithm, named Basic Iterative Method (BIM), which replaces the single-step update with multiple small-step updates. BIM performs per-pixel clipping of adversarial images, projecting adversarial images back onto the ε-neighborhood of their original images. Madry et al. [4] proposed an attack method named Projected Gradient Descent (PGD) that is similar to BIM. The difference between PGD and BIM is that PGD uses the randomly perturbed image in the ε-neighborhood of x as the initial image to start the iteration. Dong et al. [5] presented the Momentum Iterative Gradient-based Method (MIM) that adds a momentum term in the iterative process to produce adversarial examples. By accumulating the gradients in each iteration, MIM can get rid of the poor local maxima. Besides, the authors introduced a method called ensemble in logits that uses MIM to attack multiple models. To attack an ensemble of N models, they first fused the logits as

$$l(x) = \textstyle\sum_{n=1}^{N} \omega_n l_n(x), \tag{4}$$

where $l_n(x)$ are the logits of the n-th model, ω_n is the weight with $\omega_n \geq 0$ and $\sum_{n=1}^{N} \omega_n = 1$. Then, the loss function $J(x, y)$ is defined as

$$J(x, y) = -1_y \cdot log(softmax(l(x))), \tag{5}$$

where y is the ground-truth label of x, and 1_y is the one-hot encoding of y. Carlini and Wagner [6] proposed CW_0, CW_2, and CW_∞. Among the three attacks, CW_2 is the most effective and commonly used attack method, which can maintain high attack success rates and produce very tiny adversarial perturbations.

2.3 Defenses Against Adversarial Attacks

To improve the robustness of DNN models to adversarial examples, many methods have been proposed, such as defensive distillation [7], [8], gradient regularization [9], [10], adversarial training [11], [12], distributional smoothing [35], randomized models [36], [37], and verifiable defense [38], [39]. These defense methods are non-adaptive, because they often involve modifications of the architectures or the training processes, yielding an increased requirement of training examples or computational resources. There are also some studies treating adversarial perturbations as a

kind of noise, and denoising adversarial examples before they are fed into the targeted model. For instance, [15] and [16] leverage image transformations like JPEG compression, total variance minimization, bit-depth reduction to preprocess input images. [13] and [14] utilize autoencoder or UNet to reconstruct input images. However, prepropcessing input images often lead to the loss of image information and prediction accuracy.

Complimentary to the defense strategies mentioned above, an alternative line of studies focuses on picking out adversarial examples in the testing phase. [28] and [29] introduce an additional category in classifiers solely for adversarial examples, and detect adversarial examples according to the prediction of the new category. However, adding adversarial examples as an extra category requires modifying the architecture of the original classifier. Lu et al. [40] utilized a RBF-SVM classifier with discrete codes generated at high-level ReLUs in classifiers to detect adversarial samples. Metzen et al. [18] trained an auxiliary network which uses the middle-layer outputs as features to predict the probability of an input being adversarial. KD+BU [19], LID [20], and ML-LOO [21] leverage the distribution characteristics at hidden-layers in classifiers of different classes to distinguish adversarial examples. Although these strategies show compelling performances on a number of state-of-the-art adversaries, one major drawback is that they depend closely on the details of the targeted model. [22] and [23] train DNN-based binary classifiers as detectors to identify adversarial examples. Although these two methods can get rid of the dependence on the targeted model, they still need the prior knowledge of attacks. Our method does not rely on the targeted model, nor does it need any prior knowledge about attacks. So, it has good generalization and transferability, and can be reused to protect different models against different attacks.

3 Design

We present our defense, termed as Mutual Information Dual Autoencoder Detector (MIAED), in this section. The first task of our approach is to obtain valuable high dimensional features of input images, and the second task is to detect adversarial examples. These two tasks correspond to the two main steps of our approach: training an MI dual autoencoder and acquiring an adversarial example detector.

For high dimensional data like images, the autoencoder has a powerful ability to capture the high dimensional feature distributions without supervised information. Let $X = \{x_1, x_2, \cdots, x_n\}$ be the input images, $Z = \{z_1, z_2, \cdots, z_n\}$ be the corresponding latent representations, and $X' = \{x'_1, x'_2, \cdots, x'_n\}$ be the corresponding reconstructed images. The encoder E is applied to encode x_i into z_i, i.e., $z_i = E(x_i)$, and the decoder G reconstructs x_i based on z_i, i.e., $x'_i = G(z_i)$. While a good reconstruction effect of an autoencoder is critical, we prefer to obtain more discriminative latent representations. However, most of existing methods based on autoencoder endeavour to minimize the reconstruction cost. In fact, there is no substantial connection between the reconstruction cost and the discriminability of latent representations [24]. Therefore, we leverage the mutual information maximization and prior distribution matching to improve the discriminability and the generalization of the learned latent representations.

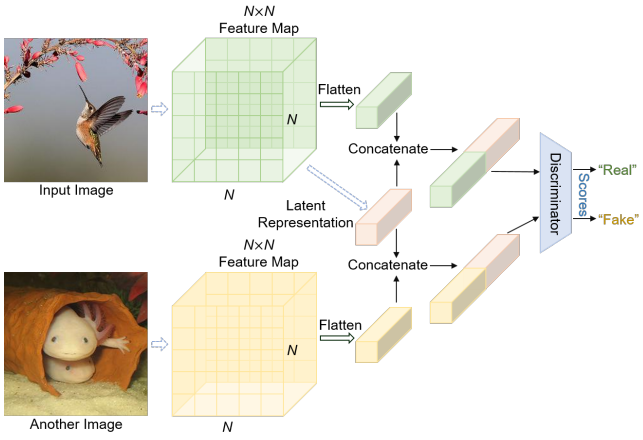


Fig. 2. Global mutual information estimation. We first concatenate the latent representation of an input image with its flattened lower-level $N \times N$ feature map to obtain a positive sample, and concatenate the same latent representation with a flattened feature map from another image to obtain a negative sample. Then, the positive and negative samples are passed into a discriminator to get scores.

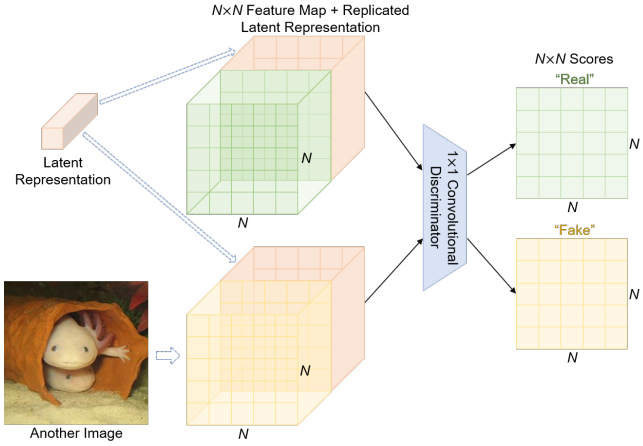


Fig. 3. Local mutual information estimation. We concatenate the latent representation of an input image with its lower-level $N \times N$ feature map at every location to form a positive sample, and use the same way as the global mutual information estimation to produce negatives. A 1×1 convolutional discriminator is utilized as the estimation network.

3.1 Generating High Dimensional Features

As shown in Fig. 1, the MI dual autoencoder includes three components: mutual information maximization, prior distribution matching and two-stream decoder. Next, we will explain these components in details.

3.1.1 *Mutual Information Maximization*

The mutual information can be used to effectively estimate the similarity between X and Z. By maximizing the mutual information between X and Z to improve the information absolute magnitude of the latent representations, and then improve the discriminability of the latent representations. In addition to Eq. (1), the mutual information can be defined as the Kullback-Leibler (KL-) divergence between

the joint, $p(z|x)p(x)$, and the product of the marginals, $p(z)p(x)$:

$$\begin{aligned} I(X,Z) &= \iint p(x,z) log \frac{p(x,z)}{p(x)p(z)} dxdz \\ &= \iint p(z|x)p(x) log \frac{p(x|z)}{p(z)} dxdz \\ &= KL(p(z|x)p(x)||p(z)p(x)), \end{aligned} \tag{6}$$

where $p(z) = \int p(z|x)p(x)dx$ (total probability formula), $p(x)$ is the distribution of the input images, $p(x,z)$ is the joint distribution of the input images and the latent representations, $p(z|x)$ is the conditional distribution of the latent representations. We follow the idea of the Mutual Information Neural Estimation (MINE) [25] to estimate mutual information, which uses a discriminator to distinguish the joint distribution and the product of marginal distributions. MINE adopts the Donsker-Varadhan representation of the KL-divergence to represent a lower-bound of MI:

$$\begin{aligned} I(X,Z) \geq \widehat{I}_{DV}(X,Z) &= \mathbb{E}_{(x,z)\sim p(z|x)p(x)}[T_\varphi(x,z)] \\ &- log \mathbb{E}_{(x,z)\sim p(z)p(x)}[e^{T_\varphi(x,z)}], \end{aligned} \tag{7}$$

where T_φ is a neural network discriminator with the parameters φ. We optimize the encoder E with the parameters ω by concurrently estimating and maximizing the mutual information:

$$(\hat{\omega}, \hat{\varphi})_{E,T} = \underset{\omega,\varphi}{argmax}\ \widehat{I}_{DV}(X,Z). \tag{8}$$

However, KL-divergence is asymmetric, i.e., $KL(p||q) \neq KL(p||q)$. As we are chiefly interested in maximizing mutual information, and not concerned with its exact value, we adopt the Jensen-Shannon (JS-) divergence MI estimator [41] instead of the KL divergence MI estimator, that is,

$$\begin{aligned} \widehat{I}_{JS}(X,Z) &= \mathbb{E}_{(x,z)\sim p(z|x)p(x)} log[T_\varphi(x,z)] \\ &+ \mathbb{E}_{(x,z)\sim p(z)p(x)} log[1 - T_\varphi(x,z)]. \end{aligned} \tag{9}$$

In practice, as shown in Fig. 2, we concatenate the latent representation of an image with its flattened lower-level $N \times N$ feature map to form a positive sample, and concatenate the same latent representation with a flattened feature map from another image to form a negative sample. We pass the concatenated samples through the discriminator T_φ to solve Eq. (9). Note that Eq. (9) is the Global Mutual Information Estimation (GMIE) that estimates the global mutual information between the input and the output of an encoder. In addition, we adopt the Local Mutual Information Estimation (LMIE) [26] to estimate the average mutual information between the latent representation and the local patches of an image. We extract the feature map $C_\omega(x) = \{C_\omega^{(i)}\}_{i=1}^{N\times N}$ from the middle layer (we select the last convolution layer of the encoder in this work) of the encoder. As shown in Fig. 3, the latent representation is concatenated with the feature map at every location, and a 1×1 convolutional discriminator is then utilized as the estimation network. The average local mutual information estimation can be defined as

$$\begin{aligned} \widehat{I}_L(X,Z) = \frac{1}{N^2} \sum_{i=1}^{N^2} &(\mathbb{E}_{(x,z)\sim p(z|x)p(x)} log[T_\varphi^L(C_\omega^{(i)}(x), z)] \\ &+ \mathbb{E}_{(x,z)\sim p(z)p(x)} log[1 - T_\varphi^L(C_\omega^{(i)}(x), z)]), \end{aligned} \tag{10}$$

where T denotes the convolutional discriminator. During the training process, the global and local mutual information estimation are used together to maximize the MI between the input and the output of the encoder.

3.1.2 Prior Distribution Matching (PDM)

A good property of a desirable latent representation is that it contains the original information as much as possible. Beyond that, we also expect that the latent representation is independently controllable and regular. To achieve this goal, we impose statistical restrictions on the latent representations, making the learned representations implicitly follow a prior distribution. Let $q(z)$ be the prior distribution that we want to impose on the latent representations. We train a discriminator D_ϕ to estimate the divergence $D(q(z)||p(z|x))$, and then minimize the estimate by training the encoder to make $p(z|x)$ approximate $q(z)$, which is similar to the adversarial autoencoder (AAE) [27],

$$\begin{aligned} (\hat{\omega}, \hat{\phi})_{E,D} &= \underset{\omega}{argmin}\ \underset{\phi}{argmax}\ \widehat{D}_\phi(q(z)||p(z|x)) \\ &= \mathbb{E}_{z\sim q(z)} log[D_\phi(z)] \\ &+ \mathbb{E}_{x\sim p(x)} log[1 - D_\phi(E_\omega(x))]. \end{aligned} \tag{11}$$

The three objectives, including global MI maximization, local MI maximization, and prior distribution matching, have the same target which is to impose restrictions on the encoder. The gradient descent algorithm is adopted to optimize the parameters of neural networks. Therefore, we add the negative sign before Eqs. (9) and (10) to change maximization to minimization. For Eq. (11), we train the encoder and the discriminator adversarially based on WGAN-GP [42]. By fusing the three objectives, we can arrive at our final objective:

$$\begin{aligned} L_e = &-\alpha(\mathbb{E}_{(x,z)\sim p(z|x)p(x)} log[T_{\varphi_1}(x,z)] \\ &+ \mathbb{E}_{(x,z)\sim p(z)p(x)} log[1 - T_{\varphi_1}(x,z)]) \\ &- \frac{\beta}{N^2} \sum_{i=1}^{N^2} (\mathbb{E}_{(x,z)\sim p(z|x)p(x)} log[T_{\varphi_2}(C_\omega^{(i)}(x), z)] \\ &+ \mathbb{E}_{(x,z)\sim p(z)p(x)} log[1 - T_{\varphi_2}(C_\omega^{(i)}(x), z)]) \\ &- \gamma(\mathbb{E}_{x\sim p(x)}(D_\phi(E_\omega(x)), \end{aligned} \tag{12}$$

where N represents the edge length of the feature map, and α, β, and γ are the hyperparameters for balancing the losses.

3.1.3 Two-Stream Decoder

The decoder, as an indispensable part of the autoencoder, is applied to reconstruct the input images based on the latent representations. We develop a two-stream decoder to reconstruct the input images. The first stream uses the original latent representations as inputs, and we leverage the L_2 norm to measure the gap between the input images and the reconstructions:

$$L_f = ||x - G(E(x))||_2, \tag{13}$$

where $||\cdot||_2$ denotes the L_2 norm, i.e., the mean square error. The second stream utilizes the noisy versions of the latent representations for reconstruction learning, where the reconstruction loss is

$$L_s = ||G(E(x)) - G(E(x) + \xi)||_2, \tag{14}$$

where ξ denotes the random Gaussian noise. Note that L_s calculates the relative reconstruction loss, i.e., the difference between the reconstruction result based on x and the reconstruction result based on the noisy version. L_s can enhance the robustness of latent representations to noise, making clean images and their adversarial versions close in the latent space.

Although the MI dual autoencoder does not use any adversarial examples in the whole training process, we can guarantee that the latent representations of an image and its adversarial version are similar. By combining Eq. (12) and the two reconstruction losses, an unified loss function is obtained. Overall, the objective of our whole autoencoder is formulated as:

$$minimize\ L_{total} = L_e + \epsilon \cdot L_f + \delta \cdot L_s. \tag{15}$$

In practice, we set the hyperparameters α and β in Eq. (12) on the same order of magnitude, and the hyperparameters ϵ and δ on the same order of magnitude. However, if we have different requirements, we can choose appropriate values for these hyperparameters according to the actual situation.

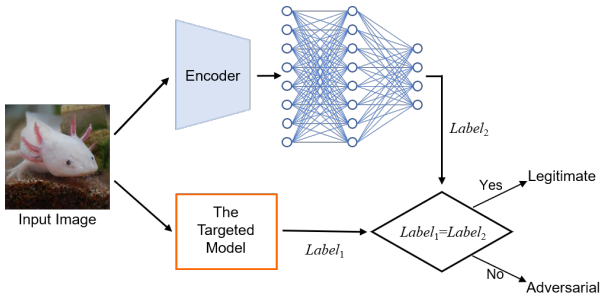


Fig. 4. Adversarial example detection. Given an input image, it is fed to the targeted model and the well-trained encoder simultaneously. The output of the encoder is used as the input of the simple fully connected neural network. By comparing the outputs of the targeted model and the simple network, we can judge whether the input image is adversarial or not.

3.2 Adversarial Example Detection

A well-trained MI dual autoencoder is considered as the initial condition of our approach. Our goal is to obtain a latent space, in which an image and its adversarial version have similar distribution information. Therefore, the decoder is useless in the stage of adversarial example detection. We use the encoder as a converter from the original space to the latent space. Although we know that an image and its adversarial version are similar in the latent space, different in the high-level space of the targeted model, we do not have reference substances for input images. In other words, if we have clean versions of all inputs, we can pick out adversarial examples by comparing the representations of inputs and their clean versions in the two spaces. This is obviously unrealistic. In this work, we train a simple fully connected neural network for the learned latent representations, which has the same output eigenspace as the targeted model. In this way, an input example is its own reference, and we only require the outputs of the targeted model without any other knowledge. As shown in Fig. 4, if the output results of the simple network and the targeted model on an image are consistent, the image is a legitimate example, and otherwise it is an adversarial example. The property independent of attack methods and the targeted model gives our method good generalization and good transferability, it can effectively resist different attacks and protect different models.

4 Performance Evaluation

In this section, we present the experimental settings, followed by the comparison between the proposed method and several state-of-the-art detection methods on three real datasets. The code of this work is available at: https://github.com/Gaoyitu/MIAED.

4.1 Experimental Settings

Datasets: We extensively evaluate our proposed method on three datasets, including MNIST, CIFAR-10, and ImageNet. MNIST is a grayscale image dataset with the image size of (28×28), including 60000 training images and 10000 testing images. CIFAR-10 contains 50000 training images and 10000 testing images with the size of $(32 \times 32 \times 3)$. For ImageNet, we select 10 categories, i.e., goldfish, ostrich, axolotl, chameleon, hummingbird, admiral, violin, ice cream, teapot, and rapeseed, from ILSVRC2012, with each category containing 1300 training images and 50 test images with the size of $(224 \times 224 \times 3)$ to test different detection methods, and use the whole ILSVRC2012 to verify the performance of our approach.

TABLE 1
The architectures of the designed classifiers for MNIST and CIFAR-10.

MNIST C1	CIFAR-10 C2
Conv(32,3,1), ReLU	(Conv(64,3,1), ReLU)×2
Max Pooling 2×2	Max Pooling 2×2
Conv(64,3,1), ReLU	(Conv(128,3,1), ReLU)×2
Max Pooling 2×2	Max Pooling 2×2
Fully Connected 200	(Fully Connected 256)×2
Softmax 10	Softmax 10

Conv(d, k, s) denotes the convolutional layer with d as dimension, k as kernel size and s as stride.

TABLE 2
Classifiers for CIFAR-10 and ImageNet.

CIFAR-10	ImageNet
VGG16 [44]	VGG16 [44]
MobileNet [45]	MobileNet [45]
ResNet50 [46]	DenseNet121 [47]
	InceptionNetV3 [48]

Classifiers: We design a classifiers for MNIST, and a classifier for CIFAR-10. The detailed architectures of the two classifiers are shown in Table 1. Other classifiers utilized in comparison are shown in Table 2. All classifiers are trained using the Adam optimizer [43] ($\beta_1 = 0.9, \beta_2 = 0.999$) with the batch size of 128, the learning rate of 0.001, and the epochs of 50.

Baseline methods: We compare our method (MIAED) with the state-of-the-art detection methods including

TABLE 3
The architectures of our MI dual autoencoders for MNIST and CIFAR-10.

MNIST		CIFAR-10	
Encoder	Decoder	Encoder	Decoder
(Conv(16, 3, 1), ReLU)×2	Fully Connected 7×7×32, ReLU	(Conv(32, 3, 1), ReLU)×2	Fully Connected 8×8×64, ReLU
Max Pooling 2×2	Reshape (7, 7, 32)	Max Pooling 2×2	Reshape (8, 8, 64)
(Conv(32, 3, 1), ReLU)×2	Up Sampling 2×2	(Conv(64, 3, 1), ReLU)×2	Up Sampling 2×2
Max Pooling 2×2	(Conv(32, 3, 1), ReLU)×2	Max Pooling 2×2	(Conv(64, 3, 1), ReLU)×2
Flatten	Up Sampling 2×2	Flatten	Up Sampling 2×2
Fully Connected 64	(Conv(16, 3, 1), ReLU)×2	Fully Connected 64	(Conv(32, 3, 1), ReLU)×2
	Conv(1, 3, 1), Sigmoid		Conv(3, 3, 1), Sigmoid

TABLE 4
The architectures of the global MI discriminator, the local MI discriminator, the prior distribution matching discriminator and the simple fully connected neural network.

Global	Local	Prior	Simple
Full Connected 256, ReLU	Conv(256, 1, 1), ReLU	Full Connected 256, ReLU	Full Connected 1024, BN, ReLU, Dropout 0.5
Full Connected 256, ReLU	Conv(256, 1, 1), ReLU	Full Connected 256, ReLU	Full Connected 512, BN, ReLU, Dropout 0.5
Full Connected 1, Sigmoid	Conv(1, 1, 1), Sigmoid	Full Connected 1	Softmax 10

BN presents batch normalization.

TABLE 5
The accuracy of different classifiers on testing sets obtained by different attacks.

Attack	MNIST	CIFAR10						ImageNet			
	C1	C2	VGG16	MN	RN50	VGG16→C2	RN50→C2	VGG16	MN	INV3	DN121
NA	0.993	0.786	0.828	0.825	0.806	0.786	0.786	0.936	0.984	0.976	0.994
FGSM	0.297	0.178	0.306	0.100	0.806	0.513	0.504	0.036	0.428	0.508	0.402
MIM	0.0	0.017	0.113	0.025	0.027	0.526	0.495	0.0	0.134	0.228	0.004
PGD	0.0	0.016	0.115	0.038	0.026	0.551	0.554	0.0	0.140	0.172	0.004
BIM	0.0	0.011	0.115	0.039	0.024	0.544	0.535	0.0	0.134	0.186	0.004
CW_2	0.032	0.091	0.097	0.045	0.091	0.695	0.704	0.042	0.010	0.018	0.006

NA means no attack. MN is MobileNet, RN50 is ResNet50, INV3 is InceptionNetV3, and DN121 is DenseNet121. VGG16→C2 denotes the accuracy of C2 on adversarial examples generated by different attacks with VGG16.

TABLE 6
The architectures of our MI dual autoencoders for ImageNet.

ImageNet	
Encoder	Decoder
(Conv(32, 3, 1), ReLU)×2	Fully Connected 14×14×128, ReLU
Max Pooling 2×2	Reshape (14, 14, 128)
(Conv(64, 3, 1), ReLU)×2	Up Sampling 2×2
Max Pooling 2×2	(Conv(128, 3, 1), ReLU)×2
(Conv(128, 3, 1), ReLU)×2	Up Sampling 2×2
Max Pooling 2×2	(Conv(128, 3, 1), ReLU)×2
(Conv(128, 3, 1), ReLU)×2	Up Sampling 2×2
Max Pooling 2×2	(Conv(64, 3, 1), ReLU)×2
Flatten	Up Sampling 2×2
Fully Connected 64	(Conv(32, 3, 1), ReLU)×2
	Conv(3, 3, 1), Sigmoid

KD+BU [19], LID [20], the grafted network detector (GND) [18] (where a temporary name is used here for convenience), the single-stream detector (SSD) [22], the two-stream detector (TSD) [23], and the convolution dual autoencoder detector (CAED). Table 3 and Table 6 show the architectures of CAED and our method for the three datasets. Table 4 shows the architectures of the global MI discriminator, the local MI discriminator, the prior distribution matching discriminator and the simple fully connected neural network.

Attack techniques: Five attack techniques, i.e., FGSM [2], BIM [3], PGD [4], MIM [5], and CW_2 [6], are employed to examine the effectiveness of our approach. For FGSM, BIM, PGD, and MIM, we set $\varepsilon = 0.3$ (see Eq. (3), out of 1.0) on MNIST, and $\varepsilon = 8/255$ on CIFAR-10 and ImageNet.

Evaluation metrics: To evaluate the effectiveness of our proposed method, we adopt the recall (Rec), precision (Prec), F1, specificity (Spec), and accuracy (Acc) to quantify the detection performance. The test data in our evaluation includes clean images, noisy images and adversarial images. Noisy images are produced by adding random Gaussian noise ($\mu = 0.0$, $\sigma = 0.1$ for MNIST, $\mu = 0.0$, $\sigma = 0.01$ for CIFAR-10 and ImageNet) to the clean images. Table 5 shows the accuracy of different classifiers on adversarial examples. The number of the test examples is determined according to the number of the samples for which the targeted model can correctly identify the clean and noisy versions, but cannot correctly identify the adversarial versions. Since the accuracy of a classifier on noisy examples is similar to that on clean examples, we can only consider clean examples to estimate the number of test samples. For example, when C1 is used as the targeted model to defend against FGSM, C1 has an accuracy of 0.993 on clean examples and 0.297 on adversarial examples, hence the number of examples for each type of test (clean, noisy and adversarial) is approximately $10000 \times (0.993 - 0.297) = 6960$. Adversarial images are positive examples, while clean and noisy images are negative examples. To balance the number of positive and negative samples, we first double the number of positive samples, and then calculate the evaluation metrics.

Parameter settings: The five detectors (GND, SSD,

TABLE 7
Comparison of recall, precision, F1, specificity and accuracy scores (%) for various adversarial detection methods when resisting white-box attacks. C1, C2, and VGG16 are the targeted models for MNIST, CIFAR-10 and ImageNet, respectively. The test adversarial examples are generated by different attacks on the targeted models.

Dataset	Method	Metric									
		PGD					CW_2				
		Rec	Prec	F1	Spec	Acc	Rec	Prec	F1	Spec	Acc
MNIST	SSD-PGD	**100**	**100**	**100**	**100**	**100**	0.0	Nan	Nan	**100**	50
	SSD-CW_2	**100**	98.96	99.48	98.95	99.47	98.28	98.97	98.62	98.97	98.63
	TSD-PGD	**100**	**100**	**100**	**100**	**100**	0.0	Nan	Nan	**100**	50
	TSD-CW_2	**100**	99.25	99.62	99.24	99.62	97.82	98.76	98.54	99.27	98.55
	GND-PGD	**100**	99.98	99.99	99.98	99.99	0.03	60.00	0.06	99.98	50.01
	GND-CW_2	1.48	25.84	2.79	95.77	48.62	98.77	96.05	97.88	95.90	97.83
	LID-PGD	57.29	86.24	68.85	90.85	74.07	32.49	78.28	45.92	90.99	61.74
	LID-CW_2	36.28	82.22	50.34	92.15	64.22	58.24	88.27	70.18	92.26	75.25
	CAED	85.54	88.15	86.83	88.50	87.02	97.15	81.35	88.55	88.87	91.63
	MIAED (ours)	93.63	98.78	96.14	98.84	96.24	**99.14**	**99.03**	**99.08**	99.02	**99.08**
CIFAR-10	SSD-PGD	98.75	98.29	98.52	98.28	98.52	0.10	5.73	0.20	98.29	49.20
	SSD-CW_2	0.17	0.42	0.24	59.57	29.87	85.01	67.77	75.42	59.58	72.29
	TSD-PGD	**98.79**	99.05	**98.92**	99.05	**98.92**	0.13	11.76	0.26	99.02	49.58
	TSD-CW_2	1.40	16.50	2.58	92.92	47.16	41.94	**85.68**	56.31	92.99	67.46
	GND-PGD	90.90	**99.72**	95.11	**99.75**	95.32	1.12	81.90	2.21	**99.75**	50.44
	GND-CW_2	26.81	39.08	31.80	58.21	42.51	89.82	69.31	79.36	58.90	75.86
	LID-PGD	70.61	79.86	74.95	82.19	76.40	22.89	56.06	32.50	82.06	52.47
	LID-CW_2	40.79	91.95	56.51	96.43	68.61	6.29	63.64	11.45	96.40	51.35
	CAED	85.82	75.81	80.51	72.62	79.22	87.61	76.44	81.64	73.00	80.30
	MIAED (ours)	88.42	77.02	82.33	73.62	81.02	**90.07**	77.41	**82.83**	74.00	**81.54**
ImageNet	SSD-PGD	41.54	66.21	51.05	78.80	60.17	13.25	38.63	19.73	78.95	46.10
	SSD-CW_2	83.73	50.00	62.61	16.27	50.00	83.91	50.06	62.71	16.31	50.11
	TSD-PGD	88.89	**99.76**	**94.01**	99.79	**94.34**	0.21	66.67	0.43	99.89	50.05
	TSD-CW_2	0.0	Nan	Nan	**100**	50.0	0.0	Nan	Nan	**100**	50.0
	GND-PGD	89.57	89.79	89.68	89.79	89.68	0.43	66.67	0.85	99.79	50.11
	GND-CW_2	54.27	49.90	52.00	45.51	49.89	91.63	62.75	74.49	45.60	68.62
	LID-PGD	67.31	94.74	78.70	96.26	81.78	3.21	46.15	6.01	96.25	49.73
	LID-CW_2	0.0	Nan	Nan	**100**	50.0	0.0	Nan	Nan	**100**	50.0
	CAED	89.10	70.03	78.42	61.86	75.48	89.51	70.08	78.61	61.78	75.64
	MIAED (ours)	**92.95**	74.49	82.70	68.16	80.56	**93.36**	**74.59**	**82.93**	68.20	**80.78**

TSD, CAED, and our method) are trained by Adam ($\beta_1 = 0.5, \beta_2 = 0.999$) with the batch size of 128, the learning rate of 0.0001, and the epochs of 100. We set $\alpha = 0.01$, $\beta = 0.01$, $\gamma = 0.001$ in Eq. (12), and $\epsilon = 1.0$, $\delta = 1.0$ in Eq. (15). For GND, SSD, and TSD, each batch consists of 64 clean images and their adversarial images. The training adversarial images are generated by C1, C2, and VGG16 for MNIST, CIFAR-10, and ImageNet, respectively. Meanwhile, C1, C2, and VGG16 are the assisted model for GND and LID.

4.2 Quantitative evaluation

For fair comparison, we strictly follow the settings described in Parameter Settings to train different detection methods in the same environment. The performance of the proposed approach is compared in seven aspects: defend against white-box attacks, defend against black-box attacks, protection of different models, detection ability under different perturbation intensity, verification on the large-scale dataset, ablation study, and robustness analysis.

4.2.1 Defend against White-box Attacks

In the experiment of defending against white-box attacks, C1, C2, and VGG16 are the targeted models for different datasets, and are used to generate all adversarial images including the training adversarial images and the test adversarial images. Table 7 shows the detection results of different defense methods when facing white-box attacks. Five attacks are adopted in this experiment. Since different methods have similar effects on defending against FGSM, MIM, PGD, and BIM, only PGD and CW_2 are selected for display. SSD-PGD represents SSD trained with the assistance of PGD, while other methods follow the same naming rule. On MNIST, SSD-PGD can accurately identify all adversarial images generated by PGD, but judges all adversarial images generated by CW_2 as clean. The perturbations produced by CW_2 are much smaller than that produced by PGD, so SSD-PGD is difficult to resist CW_2. SSD-CW_2 benefits from the training examples of CW_2, thus achieving good defense results in resisting PGD and CW_2. The defense performance of TSD is similar to that of SSD. GND-PGD can defend against PGD effectively, and GND-CW_2 can defend against CW_2 effectively. The defense performance of LID is unpleasing no matter which attack is used for training. CAED performs well when resisting PGD and CW_2. Notably, our method performs significantly better than CAED, and scores more than 99% under all metrics when defending against CW_2, which demonstrates that the latent space generated by our method is indeed better than that of CAED. Our method and CAED do not require the assistance of any attack and hence, their defense performances are not affected seriously when resisting different attacks. It is worth noting that, our method and CAED perform better against CW_2 than against PGD. The greater the difference between images, the greater the difference between their latent representations. The perturbations produced by PGD are larger than that produced by CW_2, particularly on MNIST where we set ε to 0.3. Therefore, our method and

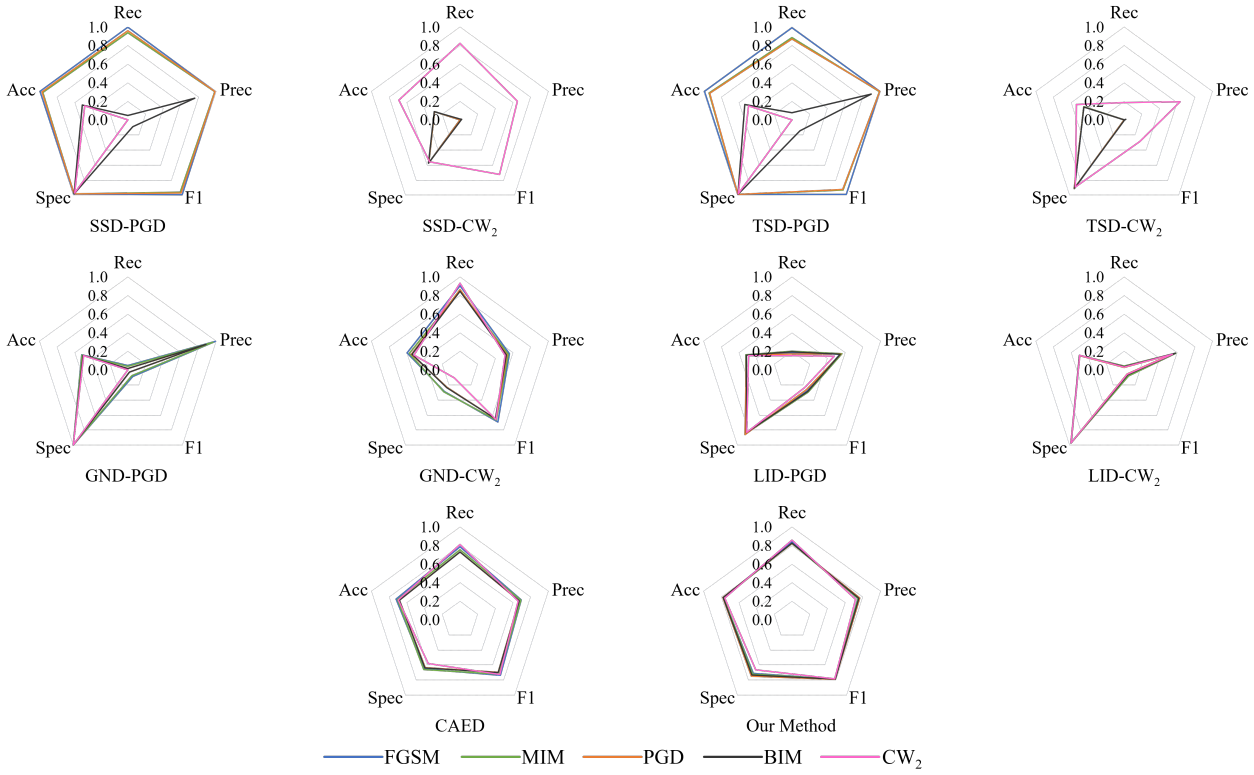


Fig. 5. Comparison of recall, precision, F1, specificity, and accuracy for detection methods when resisting black-box attacks on CIFAR-10. C2 is the targeted model, while ResNet50 is used to generate all test adversarial images with different attacks.

TABLE 8
Comparison of recall, F1, and accuracy scores (%) for various detection methods when resisting black-box attacks on CIFAR-10. C2 is the targeted model, and the test adversarial examples are generated by different attacks on VGG16 and ResNet50.

Model	Method	Metric								
		PGD			BIM			CW_2		
		Rec	F1	Acc	Rec	F1	Acc	Rec	F1	Acc
VGG16	SSD-PGD	**97.60**	98.04	98.05	0.17	0.34	49.40	0.0	Nan	48.13
	SSD-CW_2	0.0	Nan	28.77	30.51	35.18	43.78	84.26	75.11	72.08
	TSD-PGD	97.35	**98.20**	**98.21**	4.04	7.71	51.66	0.0	Nan	49.39
	TSD-CW_2	0.53	0.97	46.08	14.97	24.31	53.38	42.08	56.01	66.96
	GND-PGD	4.44	8.50	52.18	4.73	9.01	52.30	0.0	Nan	50.00
	GND-CW_2	90.26	67.27	56.09	85.55	66.44	55.27	89.94	67.70	55.18
	LID-PGD	17.85	26.91	51.52	17.38	26.40	51.54	17.00	25.05	49.12
	LID-CW_2	3.88	7.26	50.46	3.62	6.80	50.35	2.06	3.94	49.74
	CAED	80.05	75.85	74.51	81.26	77.03	75.77	82.44	76.47	74.63
	MIAED (ours)	86.11	81.94	81.03	**88.51**	**81.32**	**80.36**	**90.34**	**83.30**	**81.88**
ResNet50	SSD-PGD	**95.89**	**97.19**	**97.22**	4.77	8.98	51.62	0.0	Nan	48.70
	SSD-CW_2	0.0	Nan	28.62	0.77	1.08	29.17	82.29	72.64	69.01
	TSD-PGD	87.01	92.70	93.14	7.90	14.51	53.49	0.0	Nan	49.26
	TSD-CW_2	0.0	Nan	45.59	0.16	0.30	45.51	18.63	28.79	53.92
	GND-PGD	1.53	3.00	50.66	1.49	2.93	50.65	0.0	Nan	50.0
	GND-CW_2	86.46	65.73	54.92	82.76	64.95	54.27	85.40	66.00	51.89
	LID-PGD	17.24	26.27	51.63	19.45	28.65	51.57	15.38	23.27	49.28
	LID-CW_2	2.82	5.36	50.17	3.53	6.65	50.48	3.03	5.74	50.25
	CAED	73.10	69.82	68.40	73.25	70.01	68.63	81.13	72.73	69.58
	MIAED (ours)	82.18	79.20	78.42	**84.58**	**79.00**	**78.05**	**86.98**	**78.30**	**76.17**

CAED have better performances than the other competitors in resisting CW_2.

On CIFAR-10, we can see that SSD-PGD and TSD-PGD perform better than SSD-CW_2 and TSD-CW_2 in resisting PGD. SSD-CW_2 and TSD-CW_2 perform better than SSD-PGD and TSD-PGD in resisting CW_2. Nevertheless, SSD-CW_2 and TSD-CW_2 do not perform well against CW_2. GND and LID have the same shortcoming as SSD and TSD, that is, they have poor generalization capability and are difficult to defend against other attacks effectively after training with a certain attack. There is little performance difference for CAED in defending against different attacks. Our approach is still significantly superior to CAED, both in resisting PGD and in resisting CW_2. Note that, TSD-CW_2 has the highest precision score and GND-PGD has the highest specificity score. This does not indicate that TSD-CW_2 and GND-PGD

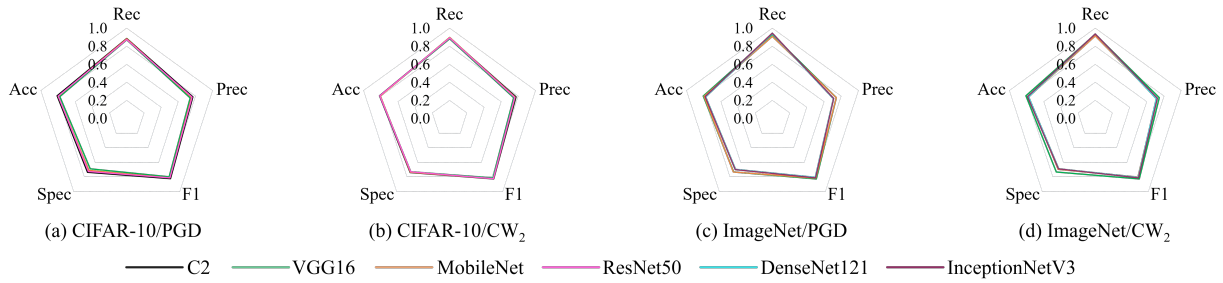


Fig. 6. Visualizing the defense capability of our method when protecting different models. C2, VGG16, MobileNet and ResNet50 are used for CIFAR-10, and VGG16, MobileNet, DenseNet121 and InceptionNetV3 are used for ImageNet. All adversarial images are generated by the protected model, e.g., when protecting C2, the adversarial examples are produced by attacks with C2. (a): Defending against PGD on CIFAR-10. (b) Defending against CW_2 on CIFAR-10. (c) Defending against PGD on ImageNet. (d) Defending against CW_2 on ImageNet.

TABLE 9
The recall, F1 and accuracy scores (%) of the proposed approach in defending against adversarial examples with different perturbation intensity on CIFAR-10.

Model	Perturbation Intensity	FGSM			MIM			PGD			BIM		
		Rec	F1	Acc	Rec	F1	Acc	Rec	F1	Acc	Rec	F1	Acc
MobileNet	2/255	85.36	75.43	72.20	87.36	78.45	76.00	87.25	78.29	75.81	87.37	78.40	75.92
	4/255	86.50	77.32	74.63	87.48	78.92	76.63	87.51	78.79	76.45	87.75	78.99	76.66
	6/255	86.82	77.84	75.28	87.43	78.96	76.70	87.41	78.81	76.49	87.57	78.95	76.65
	8/255	87.02	78.14	75.65	87.32	78.94	76.70	87.27	78.76	76.47	87.51	78.92	76.63
	10/255	87.10	78.25	75.79	87.18	78.89	76.66	87.20	78.74	76.45	87.47	78.90	76.61
ResNet50	2/255	83.83	73.71	70.09	85.92	76.73	73.94	85.87	76.73	73.96	86.02	76.84	74.07
	4/255	85.28	76.37	73.62	85.85	78.72	76.53	87.00	78.85	76.67	87.03	78.88	76.70
	6/255	85.69	77.37	74.94	86.40	78.66	76.56	86.59	78.79	76.70	86.67	78.83	76.73
	8/255	85.73	77.73	75.44	85.93	78.43	76.37	86.62	78.81	76.71	86.25	78.62	76.55
	10/255	85.38	77.71	75.51	85.36	78.12	76.10	86.30	78.66	76.59	85.86	78.40	76.35

perform better than our method. The recall of TSD-CW_2 is 41.94%, and the recall of GND-PGD is 1.12%, which means that TSD-CW_2 identifies most of the test images as clean, and GND-PGD identifies almost all test images as clean.

On ImageNet, we can see that SSD has the difficulty of defending against large-scale adversarial images. TSD-PGD performs well in resisting PGD, but still has trouble in defending against CW_2. TSD-CW_2 has no defense ability against those attacks. The generalization ability of GND is still limited, and the performance of LID is again poor. The defense capabilities of CAED and our approach are very stable, and even better than the defense effects on CIFAR-10. SSD, TSD, GND, and LID are very unstable, they are difficult to defend against different attacks effectively. CAED and our method are stable, showing similar performances regardless of which attack is considered. We can also see that MIAED is superior to CAED in all metrics. Overall, CAED and our approach have good generalization capability and can effectively defend against different attacks, while our approach is the better performer.

4.2.2 Defend against Black-box Attacks

Due to the poor effect of black-box attacks on ImageNet under the settings of our experiment, we only compare and analyse different detection methods on CIFAR-10. C2 is the targeted model, which is used to generate all training adversarial images. Meanwhile, C2 is the auxiliary model of GND and LID. The test adversarial images are produced by VGG16 and ResNet50. Table 8 shows the defense effects of different detection methods against black-box attacks. Since different methods have similar effects on defending against FGSM, MIM, and PGD, we only display PGD, BIM, and CW_2. The defense effects of the detection methods are measured by recall, F1, and accuracy in Table 8. It can be seen that different detection methods have similar performances in defending against the adversarial samples generated by VGG16 and ResNet50. SSD-PGD and TSD-PGD are effective against PGD, but not against BIM and CW_2. The defense performance of SSD-CW_2 against CW_2 is much better than that against PGD and BIM. TSD-CW_2 performs poorly against all attacks. GND-CW_2 is better than GND-PGD. LID is the opposite of GND, and LID-PGD is better than LID-CW_2. CAED and our approach are still robust, and can effectively defend against adversarial samples generated by different models. Our method significantly outperforms CAED by more than 5% across almost all metrics. Fig. 5 shows the performances of different detection approaches in resisting black-box attacks, where all test adversarial examples are generated by different attacks on ResNet50. We can intuitively see that CAED and our method are robust and can effectively defend against all attacks. We can also see that the radar figures of the proposed method are significantly larger than that of CAED.

4.2.3 Protection of different models

Recall that our approach does not require the prior knowledge of attacks and the targeted model in the whole training process. Therefore, we examine the generalization ability and the transferability of the proposed method on protecting different models. Fig. 6 shows the defense performances of our approach when protecting different models on CIAFR-10 and ImageNet. All test adversarial samples are

TABLE 10
Comparison of recall, precision, F1, specificity and accuracy scores (%) for various adversarial detection methods on MNIST. C1 is the targeted model, and the test adversarial examples are generated by different attacks on C1.

Method	Metric									
	FGSM					CW_2				
	Rec	Pre	F1	Spec	Acc	Rec	Pre	F1	Spec	Acc
CAED	81.43	73.82	77.44	85.56	84.18	97.15	81.35	88.55	88.87	91.63
CAED+GMIE	83.46	95.16	88.06	96.67	88.06	97.89	97.65	97.77	97.64	97.77
CAED+LMIE	81.07	90.95	84.43	92.16	85.47	96.96	94.48	95.71	94.34	95.65
CAED+PDM	81.17	94.66	85.63	95.59	86.88	97.41	96.85	97.13	96.83	97.12
CAED+GMIE+LMIE	84.05	95.17	88.18	97.17	90.37	98.76	98.27	98.51	98.26	98.51
CAED+GMIE+PDM	85.33	**96.75**	89.18	97.34	92.33	99.06	98.06	98.56	98.04	98.55
CAED+LMIE+PDM	82.21	94.97	88.13	95.64	88.93	97.59	96.98	97.29	96.96	97.28
CAED+GMIE+LMIE+PDM	**88.84**	96.72	**92.61**	**98.49**	**95.27**	**99.14**	**99.03**	**99.08**	**99.02**	**99.08**

TABLE 11
The recall, F1 and accuracy scores (%) of our approach in defending against adversarial examples on the whole ImageNet.

Model	Attack	Rec	F1	Acc
VGG16	FGSM	98.95	83.46	80.38
	MIM	98.96	83.56	80.53
	PGD	98.89	83.55	80.53
	BIM	98.93	83.57	80.55
ResNet50	FGSM	98.84	82.99	79.75
	MIM	98.93	83.26	80.11
	PGD	98.95	83.02	79.76
	BIM	98.93	83.26	80.11

generated by different attacks with the protected models. Fig. 6 (a) and (b) are the results of our method against PGD and CW_2 on CIFAR-10, respectively. Fig. 6 (c) and (d) are the results of our method against PGD and CW_2 on ImageNet, respectively. We can see that the scores of our method are very stable on all metrics in all cases of datasets and protected models. In summary, our proposed approach does not require the prior knowledge of attacks and the assistance of the targeted models. Therefore, our method has both good generalization and good transferability, it can defend against different attacks and can be reused to protect different models.

4.2.4 Detection ability under different perturbation intensity

To test the detection ability of the proposed method under different perturbation intensity, we set $\varepsilon = 2/255$, $4/255$, $6/255$, $8/255$ and $10/255$ in Eq. (3) for CIFAR-10. Table 9 shows the performances of our approach under different perturbation intensity, and we can see that our approach is very stable on all metrics. Regardless of the perturbation intensity, as long as the generated adversarial examples adhere to the principle that they are similar to their clean versions but can fool the targeted model, our approach can effectively capture them. As we mentioned in Section 4.1, the number of the test examples in our experiment is determined according to the number of the samples for which the targeted model can correctly identify the clean and noisy versions, but cannot correctly identify the adversarial versions. Therefore, although the attack success rates of attacks under different perturbation intensity are different, the performances of our approach shown in Table 9 are similar.

4.2.5 Verification on the large-scale dataset

To verify the performance of our approach on the large-scale dataset, we test our method on the whole ImageNet. VGG16 and ResNet50 are adopted as the targeted models, and are used to produce adversarial examples. We use an encoder similar to the feature extraction part of VGG16, and adopt the inversion of the encoder as the decoder. The dimension of the latent space is 1024. The performance of our approach is measured by recall, F1 and accuracy in Table 11. The whole ImageNet dataset has 1000 categories, which means that the simple classifier and the targeted model classify the same adversarial examples into the same categories may happen with a low frequency. Therefore, we can see that our approach has excellent recall scores. Meanwhile, we can also see that our method has good F1 and accuracy scores, this shows that our method can effectively distinguish between benign examples and adversarial examples on the whole ImageNet.

4.2.6 Ablation study

To capture the difference in contribution of the global mutual information estimation loss (Eq. (9)), the local mutual information estimation loss (Eq. (10)) and the prior distribution matching loss (Eq. (11)), we combine the CAED with every possible combination of the three losses. Table 10 shows the performances of different strategies. It clearly demonstrates that each loss can improve the performance of CAED, and the defense has the best performance when the three losses work together. Among the three losses of GMIE, LMIE and PDM, GMIE has the greatest contribution for our proposed approach, followed by PDM, LMIE contributes the least to the defense effect.

4.2.7 Robustness analysis

Although our method has achieved good performance in defending against attacks in different settings, we still have some concerns about the security of our approach when the knowledge of MIAED is leaked. To verify the robustness of our method, we adopt the ensemble attack (ensemble in logits) [5] to optimize adversarial perturbations using both MIAED and the targeted model. Fig. 7 shows the accuracy of different models on adversarial examples generated by ensemble attacks. Fig. 7 (a) shows the accuracy of VGG16 and MIAED on adversarial examples generate by different ensemble attacks using VGG16 and MIAED, and Fig. 7 (b) shows the accuracy of ResNet50 and MIAED on adversarial examples generate by different ensemble attacks using

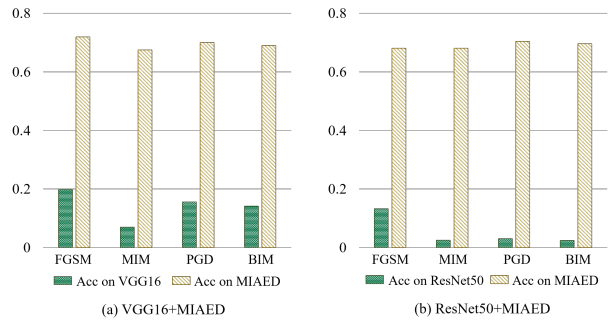


Fig. 7. The accuracy of different models when facing ensemble attacks on CIFAR-10 in white-box settings. (a): Ensemble attacks generate adversarial examples using VGG16 and MIAED. (b): Ensemble attacks generate adversarial examples using ResNet50 and MIAED.

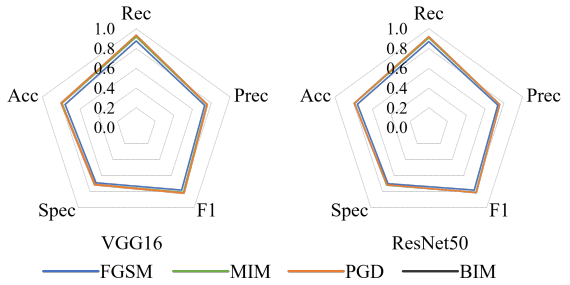


Fig. 8. The recall, precision, F1, specificity and accuracy for our method when defending against ensemble attacks on CIFAR-10.

ResNet50 and MIAED. We can see that MIAED can maintain a high classification accuracy when facing ensemble attacks. Fig. 8 shows the performances of our approach in defending against ensemble attacks. We can intuitively see that our method can effectively defend against all ensemble attacks on both VGG16 and ResNet50.

5 Conclusions

In this paper, we propose a novel approach named MIAED to detect adversarial examples. Our method, which is based on the dual autoencoder architecture, leverages the mutual information maximization and the prior distribution matching to project images to a latent space. In this space, the distances between different images will be increased, while the relative distances between similar images will be decreased. We then utilize a simple neural network to project the latent representations into an eigenspace which is the same as the output eigenspace of the targeted model. Given an input image, we can judge whether it is adversarial by comparing its outputs of the simple network and the targeted model. The proposed approach does not rely on any prior knowledge of attacks, but has good generalization ability on defending against new attacks. Meanwhile, our approach does not require the assistance of the targeted model or any similar models, meaning that it has good transferability and can be reused to protect different models after once training. We evaluate our method in six scenarios: defense against white-box attacks, defense against black-box attacks, protection of different models, detection ability under different perturbation intensity, verification on the large-scale dataset, and robustness analysis. The experimental results show that our approach is very stable and can provide effective protection for different models against different attacks.

One limitation of our approach is the performance of the simple fully connected neural network, which can not effectively mine the information contained in the latent representations. Our experiments show that although the simple network can classify the vast majority of clean examples and their adversarial versions into the same categories, the classification accuracy is still 5% to 10% lower than that of the original classifier, resulting in an increase of the false alarm rate. In our future work, we will explore more effective methods to take full advantage of the latent representations of images, and improve the robustness of the targeted model itself.

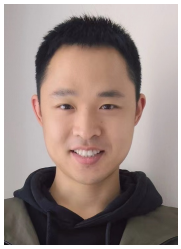
Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62162067 and 62101480, in part by the Yunnan Province Science Foundation under Grant No.202005AC160007, No.202001BB050076, and Research and Application of Object Detection based on Artificial Intelligence.

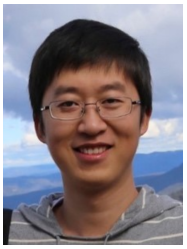
References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Daly, "Intriguing properties of neural networks", in *Proc. Int. Conf. Learn. Represent.*, 2014.
[2] I. Goodfellow, J. Shlen and C. Szegedy, "Explaining and harnessing adversarial examples", in *Proc. Int. Conf. Learn. Represent.*, 2015.
[3] A. Kurakin, I. Goodfellow and S. Bengio, "Adversarial examples in the physical world", in *Proc. Int. Conf. Learn. Represent.*, 2017.
[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards deep learning models resistant to adversarial attacks", in *Proc. Int. Conf. Learn. Represent.*, 2018.
[5] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu and J. Li "Boosting adversarial attacks with momentum", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
[6] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks", in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 39-57.
[7] N. Papernot, P. McDaniel, X. Wu, S. Jha and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks", in *Proc. IEEE Symp. Security Privacy*, 2016, pp. 582-597.
[8] N. Papernot, and P. McDaniel, "Extending defensive distillation", *arXiv preprint arXiv:1705.05264*, 2017.
[9] C. Lyu, K. Huang and H-N. Liang, "A unified gradient regularization family for adversarial examples", in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 301-309.
[10] A. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients", in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1660-1669.
[11] C. Xie, Y. Wu, L. Maaten, A. Yuille and K. He, "Feature denoising for improving adversarial robustness", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 501-509.
[12] C. Song, K. He, J. Lin, L. Wang and J. Hopcroft, "Robust local features for improving the generalization of adversarial training", in *Proc. Int. Conf. Learn. Represent.*, 2020.
[13] X. Jia, X. Wei, X. Cao and H. Foroosh, "ComDefend: An efficient image compression model to defend adversarial examples", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6084-6092.
[14] S. Gao, S. Yao and R. Li, "Transferable adversarial defense by fusing reconstruction learning and denoising learning", in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2021, pp. 1-6.

[15] C. Guo, M. Rana, M. Cisse and L. Maaten, "Countering Adversarial Images Using Input Transformations", in *Proc. Int. Conf. Learn. Represent.*, 2018.
[16] A. Bhagoji, D. Cullina, C. Sitawarin and P. Mittal, "Enhancing Robustness of Machine Learning Systems via Data Transformations", in *Proc. Ann. Conf. Inform. Scienc. Syst.*, 2018.
[17] J. Lu, T. Issaranon and D. Forsyth, "SafetyNet: Detecting and rejecting adversarial examples robustly", in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 446-454.
[18] J. Metzen, T. Genewein, V, Fischer and B. Bischoff, "On detecting adversarial perturbations", in *Proc. Int. Conf. Learn. Represent.*, 2017.
[19] R. Feinman, R. Curtin, S. Shintre and A. Gardner, "Detecting adversarial samples from artifacts", *arXiv preprint arXiv:1703.00410*, 2017.
[20] X. Ma, B. Li, Y. Wang, S. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. Houle and J. Bailey, "Characterizing adversarial subspaces using local intrinsic dimensionality", in *Proc. Int. Conf. Learn. Represent.*, 2018.
[21] P. Yang, J. Chen, C. Hsieh, J. Wang and M. Jordan, "Ml-loo: Detecting adversarial examples with feature attribution", in *Proc. AAAI Conf. Artif. Intell.*, 2020.
[22] Z. Gong, W. Wang and W. Ku, "Adversarial and clean data are not twins", *arXiv preprint arXiv:1704.04960*, 2017.
[23] S. Gao, S. Yu, L. Wu, S. Yao and X. Zhou, "Detecting adversarial examples by additional evidence from noise domain", *IET Image Process.*, vol. 16, pp. 378-392, 2022.
[24] X. Yang, C. Deng, F. Zheng, J. Yan and W. Liu, "Deep spectral clustering using dual autoencoder network", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4061-4070.
[25] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville and R. D. Hjelm, "Mutual information neural estimation", in *Proc. Int. Conf. Mach. Learn.*, 2018.
[26] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler and Y. Bengio, "Learning deep representations by mutual information estimation and maximization", in *Proc. Int. Conf. Learn. Represent.*, 2019.
[27] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow and B. Frey, "Adversarial autoencoders", in *Proc. Int. Conf. Learn. Represent. Workshop Track*, 2016.
[28] K. Grosse, P. Manoharan, N. Papernot, M. Backes and P. McDaniel, "On the (statistical) detection of adversarial examples", *arXiv preprint arXiv:1702.06280*, 2017.
[29] H. Hosseini, Y. Chen, S. Kannan, B. Zhang and R. Poovendran, "Blocking transferability of adversarial examples in black-box learning systems", *arXiv preprint arXiv:1703.04318*, 2017.
[30] Y.-I. Moon, B. Rajagopalan and U. Lall, "Estimation of mutual information using kernel density estimators", *Physical Review E*, vol. 52, no. 3, pp. 2318-2321, 1995.
[31] G. A. Darbellay and I. Vajda, "Estimation of the information by an adaptive partitioning of the observation space", *IEEE Trans. Inform. Theory*, vol. 45, no. 4, pp. 1315-1321, 1999.
[32] N. Kwak and C. Choi, "Input feature selection by mutual information based on Parzen window", *IEEE Trans. Patt. Analys. Mach. Intell.*, vol. 24, no. 12, pp. 1667-1671, 2002.
[33] M. M. Van Hulle, "Edgeworth approximation of multivariate differential entropy", *Neural comput.*, vol. 17, no. 9, pp. 1903-1910, 2005.
[34] T. Suzuki, M. Sugiyama, J. Sese and T. Kanamori, "Approximating mutual information by maximum likelihood density ratio estimation", in *New Chall. Feat. Select. Data Min. Knowl. Discov.*, pp. 5–20, 2008.
[35] T. Miyato, S. Meada, M, Koyama, K, Nakae and S. Ishii, "Distributional smoothing with virtual adversarial training", in *Proc. Int. Conf. Learn. Represent.*, 2016.
[36] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu and S. Jana, "Certified robustness to adversarial examples with differential privacy", in *Proc. IEEE Symp. Security Privacy*, 2019.
[37] X. Liu, M. Cheng, H. Zhang and C. Hsieh, "Towards robust neural networks via random self-ensemble", in *Proc. Europ. Conf. Comput. Vis.*, 2018, pp. 381-397.
[38] E. Wong and J. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope", in *Proc. Int. Conf. Mach. Learn.*, 2018.
[39] K. Dvijotham, S. Gowal, R. Stanforth, R. Arandjelovic, B. O'Donoghue, J. Uesato and P. Kohli, "Training verified learners with learned verifiers", *arXiv preprint arXiv:1805.10265*, 2018.
[40] J. Lu, T. Issaranon and D. Forsyth, "SafetyNet: Detecting and rejecting adversarial examples robustly", in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 446-454.
[41] S. Nowozin, B. Cseke and R. Tomioka, "f-GAN: Training generative neural samplers using variational divergence minimization", in *Proc. Conf. Neural Inf. Process. Syst.*, 2016, pp. 271-279.
[42] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin and A. Courville, "Improved training of Wasserstein GANs", in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5767–5777.
[43] D. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *Proc. Int. Conf. Learn. Represent.*, 2015.
[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", in *Proc. Int. Conf. Learn. Represent.*, 2015.
[45] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", *arXiv preprint arXiv:1704.04861*, 2017.
[46] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770-778.
[47] G. Huang, Z. Liu, L. van der Maaten and K. Q. Weinberger, "Densely connected convolutional networks", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700-4708.
[48] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818-2826.



Song Gao is currently a Postdoctoral Researcher with the National Pilot School of Software, Yunnan University, Kunming, China. He received his B.E. degree in software engineering from China University of Geosciences, his M.E. degree in technology of computer application from Yunnan University, and his Ph.D. degree in information and communication engineering from Yunnan University. His research interests include social computing, deep learning and computer vision.



Ruxin Wang is currently an associate professor with the National Pilot School of Software, Yunnan University, Kunming, China. He received his BEng from Xidian University, his MSc from Huazhong University of Science and Technology, and his PhD degree from the University of Technology Sydney. His research interests include image restoration, deep learning, and computer vision. He focuses on the topic of image synthesis by using both discriminative models and generative models. He has authored and coauthored 20+ research papers including IEEE TNNLS, TIP, TCyb, ICCV, and AAAI. He has received "the 1000 Talents Plan for Young Talents of Yunnan Province" award.

Xiaoxuan Wang is currently a lecture with the school of information science and technology, Yunnan Normal University. She received her B.E. degree in computer science and technology from Minzu University of China in 2014, and her Ph.D degree in information and communication engineering from Yunnan University in 2021. Her current research interests include spatial database, spatial data mining and big data.

Shui Yu is currently a Professor with the School of Computer Science, University of Technology Sydney, Sydney, NSW, Australia. He has authored or coauthored three monographs and edited two books, more than 350 technical papers, including top journals and top conferences, which include IEEE TPDS, TC, TDSC, TIFS, TMC, TKDE, TETC, ToN, and INFOCOM. His research interests include big data, security and privacy, networking, and mathematical modelling. He is currently with a number of prestigious Editorial Boards, including the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS as the Area Editor, IEEE Communications Magazine, and IEEE INTERNET OF THINGS JOURNAL. He is a Member of AAAS and ACM, and a Distinguished Lecturer of IEEE Communication Society.

Yunyun Dong is currently a lecturer with the National Pilot School of Software, Yunnan University. She received her B.S. degree in Network engineering from the Yunnan University in 2011, and her M.S. degree in system analysis and integration from the school of software, Yunnan University in 2014. Her current research interests include Big data indexing, distributed computing, image steganography.

Wei Zhou received the Ph.D. degree from the University of Chinese Academy of Sciences. He is currently a Full Professor with the Software School, Yunnan University. His current research interests include the distributed data intensive computing and bioinformatics. He is currently a Fellow of the China Communications Society, a member of the Yunnan Communications Institute, and a member of the Bioinformatics Group of the Chinese Computer Society. He won the Wu Daguan Outstanding Teacher Award of Yunnan University in 2016, and was selected into the Youth Talent Program of Yunnan University in 2017. Hosted a number of National Natural Science Foundation projects.

Shaowen Yao received the B.S. and M.S. degrees in telecommunication engineering from Yunnan University, Kunming, China, in 1988 and 1991, respectively, and the Ph.D. degree in computer application technology from the University of Electronic Science and Technology of China, Chengdu, China, in 2002. He is currently a Professor with the School of Software, Yunnan University. His current research interests include neural network theory and applications, cloud computing, and big data computing.