Hindawi
Security and Communication Networks
Volume 2020, Article ID 3932584, 12 pages
https://doi.org/10.1155/2020/3932584

Research Article

The Defense of Adversarial Example with Conditional Generative Adversarial Networks

Fangchao Yu,[1] Li Wang,[1] Xianjin Fang,[1] and Youwen Zhang[2]

[1]*School of Computer Science and Engineering, Anhui University of Science and Technology, Huainan 232000, China*
[2]*School of Computer Science and Engineering, Anhui University, Hefei 230601, China*

Correspondence should be addressed to Li Wang; liwang@aust.edu.cn

Received 9 February 2020; Accepted 6 May 2020; Published 25 August 2020

Academic Editor: Xiaolong Xu

Deep neural network approaches have made remarkable progress in many machine learning tasks. However, the latest research indicates that they are vulnerable to adversarial perturbations. An adversary can easily mislead the network models by adding well-designed perturbations to the input. The cause of the adversarial examples is unclear. Therefore, it is challenging to build a defense mechanism. In this paper, we propose an image-to-image translation model to defend against adversarial examples. The proposed model is based on a conditional generative adversarial network, which consists of a generator and a discriminator. The generator is used to eliminate adversarial perturbations in the input. The discriminator is used to distinguish generated data from original clean data to improve the training process. In other words, our approach can map the adversarial images to the clean images, which are then fed to the target deep learning model. The defense mechanism is independent of the target model, and the structure of the framework is universal. A series of experiments conducted on MNIST and CIFAR10 show that the proposed method can defend against multiple types of attacks while maintaining good performance.

1. Introduction

Deep learning [1–5] is a hierarchical machine learning method involving multilevel nonlinear transformations and is good at mining abstract and distributed feature representations from raw data. Deep learning can solve many problems that are considered challenging in machine learning. Recently, driven by the emergence of big data and hardware acceleration, deep learning has made significant progress in numerous machine learning domains, such as computer vision, natural language processing, edge computing [6–10], and services computing [11–13], and promotes the large-scale application of artificial intelligence technology in the real world. While deep learning has achieved great success, its performance and applications are also questioned due to the lack of interpretability [14], which means that we cannot reasonably explain the decisions made by deep learning models. This exposes deep learning-based artificial intelligence applications to potential security risks.

Many types of research have shown that deep learning is threatened by multiple attacks, such as membership inference attack [15, 16] and attribute inference attack [17]. The most serious security threat to deep learning is the adversarial example [18] proposed by Szegedy in 2013. An adversary can add small-magnitude perturbations to inputs, which can easily fool a well-performed deep learning model with few perturbations imperceptible to humans [19]. The disturbed inputs are called adversarial examples, and they make the target model report high confidence in incorrect predictions. Moreover, recent research shows that artificial intelligence applications in the real world can be exposed to adversarial samples [20], for example, attacks in the face recognition system [21] and vision system in autonomous cars [22].

With the in-depth study of adversarial examples, the development of this field mainly presents the following main trends. (1) A growing number of methods for constructing adversarial examples are proposed. According to adversarial

specificity, we can divide these attack methods into targeted attacks and nontargeted attacks. For targeted attacks, the adversary can submit well-designed inputs to the target model and cause maliciously chosen target outputs, such as $R+\mathrm{LLC}$ [23], JSMA [24], EAD [25], and C&W [26]. For nontargeted attacks, the adversary can cause the target model to misclassify well-designed inputs into classes that are different from the ground truth, such as FGSM [27], BIM [20], PGD [28], and DeepFool [29]. Even worse, the robustness of adversarial examples constantly increases, and detection and defense are challenging. (2) The cost of constructing adversarial examples is decreasing. Due to the transferability [30] of the adversarial example, the adversary can successfully launch an attack without background knowledge about the target model. (3) The range of attacks is also expanding. Adversarial examples can also successfully attack different deep learning models such as reinforcement learning models and recurrent neural network models. Moreover, attack scenarios are not limited to the computer vision. The same security risks exist in text [31] and speech [32]. Therefore, building an effective defense mechanism against adversarial examples is crucial in deep learning.

There is no uniform conclusion on the cause of the adversarial examples; thus, building a defense mechanism is challenging. In general, there are two classes of approaches to defend against adversarial examples: (1) making deep neural networks more robust by adjusting learning strategies, such as adversarial training [27, 33] and defensive distillation [34]; (2) detecting adversarial examples or eliminating adversarial noise after deep neural networks are built, such as LID [35], Defense-GAN [36], MagNet [37], and ComDefend [38]. There are some bottlenecks in these defense mechanisms. First, some defense mechanisms are only effective against specific attacks. For example, defensive distillation is effective for gradient-based attacks, and it is defeated by C&W attacks. Second, some methods require large samples and high computational costs, which limit the application scenarios for these defense mechanisms. Third, the difference between the adversarial example and the clean example is small; thus, it is difficult for current detection methods to distinguish them with high confidence. In summary, we hope to find a defense mechanism with good performance on most attacks and low computational cost.

Our work has made some progress toward building a better defense mechanism against adversarial examples in computer vision. The main reason for adversarial examples to mislead the target model is that the added noise changes the characteristics of the original inputs; thus, an intuitive approach is to remove the noise from the adversarial examples and generate a mapping of the adversarial examples to the clean examples. In computer vision, this can be posed as "translating" an input image (adversarial example) into a corresponding output image (clean example). In this paper, we use the framework proposed by Isola et al. [39] as a defense mechanism. Based on conditional adversarial networks (conditional GANs) [40], the framework consists of a generator network to translate the adversarial images to the clean images and a discriminator network to ensure that the generated images are realistic. Our method can effectively eliminate adversarial perturbations and restore the characteristics of the original clean images. The overview of the defense model is shown in Figure 1. The advantages of our method are listed as follows:

(1) The proposed method is a general-purpose defense framework. On the one hand, the defense mechanism processes the input and is model independent, which means that the target model does not need to be retrained. On the other hand, the network structure of the defense framework is based on a general-purpose solution of image-to-image, and we can apply the framework for different scenarios with only a few adjustments.

(2) Our method is simple and easy to use, and it is effective against most commonly considered attack strategies, such as FGSM, DeepFool, JSMA, and CW. Moreover, this defense mechanism shows certain transferability, which means the defense mechanism built for the specific target model is still effective for other models.

The remainder of the paper is as follows. We introduce some related works about adversarial example in Section 2. In Section 3, we review the necessary theories and concepts about adversarial example and conditional GANs. We give a detailed technical development about the framework of the generation and defense of adversarial example in Section 4. Section 5 describes the experimental results, and Section 6 concludes the paper.

2. Related Works

In this section, we introduce the application of GANs in the field of adversarial examples: generating adversarial examples with GANs and defending adversarial examples with GANs.

2.1. Generating Adversarial Examples with GANs. Xiao [41] proposed AdvGAN to generate adversarial examples. AdvGAN takes a clean image x as the input of the generator G and obtains the adversarial images as $x+G(x)$. The adversarial examples generated by AdvGAN perform high attack success rates in both semiwhite box and black-box attacks. Song et al. [42] designed an unrestricted approach to generate adversarial examples with an auxiliary classifier generative adversarial network (AC-GAN) [43]. Different from perturbation-based attacks, this approach constructs adversarial examples entirely from scratch instead of perturbing an existing data point. In addition, the adversary can specify the style of the generated adversarial examples and labels that are misclassified on the target model. Zhao et al. [44] noticed that the adversarial perturbations are often unnatural and not semantically meaningful. He proposed a framework consisting of a WGAN [45] and an inverter. The inverter maps a clean image to random dense vectors z. The generator of the WGAN obtains the $\widetilde{z}$ (perturbing z) as the input. The goal of the generator is to synthesize an image that is as close to the original image as possible. This method can

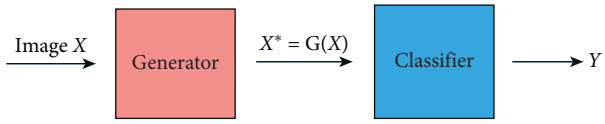

Figure 1: The overview of defense model for the adversarial example proposed in this paper. Between the input images and the classifier, we add a generator, which can eliminate the adversarial perturbations in the input images and map the adversarial examples into clean images.

generate natural and legible adversarial examples that lie on the data manifold. Hu and Tan [46] focused on adversarial examples in traditional security scenarios. They proposed the MalGAN to generate adversarial malware examples, which are able to bypass black-box machine learning-based detection models.

2.2. Defense Adversarial Examples with GANs. Lee et al. [47] introduced a novel adversarial training framework named generative adversarial trainer (GAT). The framework consists of a generator and a classifier. The generator attempts to generate adversarial perturbations that can easily fool the classifier and the classifier attempts to correctly classify both original and generated adversarial images. This approach can improve the robustness of the model and outperforms other adversarial training methods using a fast gradient method. Santhanam and Grnarova [48] proposed *cowboy*, an approach to defend against adversarial attacks with GANs. This work shows that adversarial samples lie outside of the data manifold learned by a GAN that has been trained on the same dataset. They used the discriminator (GAN) to detect adversarial examples and the generator (GAN) to eliminate adversarial perturbations. Samangouei et al. [36] proposed a new framework named Defense-GAN, which leverages the expressive capability of generative models (WGAN) to defend against adversarial examples. Defense-GAN finds a close input to the adversarial examples and sends the input to the generator of WGAN. Then, the generated images are fed to the target model.

3. Background

In this section, we introduce four methods of generating adversarial examples. In addition, GAN and its connection to our method will be discussed.

3.1. Generating Adversarial Example. The main idea of generating adversarial samples is to add appropriate perturbations to the input samples to make the noisy samples as similar to the original input as possible, but mislead the target model. We can briefly describe this process: for a given input image x, the adversary needs to find a minimal perturbation η and craft the noisy example as $x^* = x + \eta$. In recent years, many methods of generating adversarial examples have been proposed. Here, we introduce some of the most well-known attacks.

3.1.1. Fast Gradient Sign Method (FGSM) [27]. Szegedy et al. first introduced adversarial examples against deep neural networks and proposed the method named L-BFGS [18] to generate adversarial examples; however, it was time-consuming and impractical. In 2014, Goodfellow et al. argued that the primary cause of neural networks' vulnerability to adversarial perturbations is their linear nature. Based on this explanation, they proposed a simple and fast method to generate adversarial samples, named fast gradient sign method (FGSM). Let θ be the parameters of a target model, x is the input to the model, y is the label associated with x, and $J(\theta, x, y)$ is the cost function used to train the model. The adversarial sample is generated as

$$x^* = x + \varepsilon sign\left(\nabla_x J(\theta, x, y)\right), \tag{1}$$

where ε is a parameter that determines the perturbation size.

3.1.2. DeepFool [29]. FGSM is simple and effective; however, it causes a large degree of perturbations to inputs. Moosavi-Dezfooli et al. observed that adding noise along the vertical direction of the closest decision boundary to the inputs can ensure that the added perturbation is optimal. They used an iterative method to approximate the perturbation by considering that f is linearized around x_i at each iteration. The minimal perturbation is computed as

$$\begin{aligned} &\text{argmin}_{\eta_i} \|\eta_i\|_2, \\ &\text{s.t.} f(x_i) + \nabla f(x_i)^T \eta_i = 0, \end{aligned} \tag{2}$$

where η_i is the distance to the decision boundary.

3.1.3. Jacobian-Based Saliency Map Attack (JSMA) [24]. The previous two attack methods are both nontargeted attacks. Papernot et al. observed that different input features have different degrees of influence on the output of the target model. If we find that some features correspond to a specific output in the target model, we can make the target model produce a specified type of output by enhancing these features in the inputs. Based on this idea, they proposed a simple iterative method for targeted attack named the Jacobian-based saliency map attack (JSMA). First, the JSMA requires the calculation of the forward derivative, which shows the influence of each input feature on the output. Then, it can generate the adversarial saliency map and use the adversarial saliency map to find the input features that have the greatest impact on the specific output of the target model. Finally, a small perturbation added to the features can fool the neural network.

3.1.4. Carlini and Wagner (C&W) [26]. Carlini et al. proposed a method of generating a more robust adversarial example that can bypass many advanced defense mechanisms. This method treats the adversarial example as a variable, and two conditions need to be met for the attack to succeed. First, the difference between the adversarial example and the corresponding clean sample should be as small as possible. Second, the adversarial example should

make the model classification error rate as high as possible. There are three attacks for the L_0, L_1, and L_2 distance metrics, and we provide a brief description of the L_2 attack:

$$\min_w \left|\frac{1}{2}(\tanh(w)+1)\right|_2 + c \cdot g\left(\frac{1}{2}\tanh(w)+1\right). \tag{3}$$

The loss function g is defined as

$$g(x) = \max\left(\max_{i\neq l'}(Z(x)_i) - Z(x)_{t'} - \kappa\right), \tag{4}$$

where Z denotes the SoftMax function, κ is a constant used to control the confidence (as κ increases, the adversarial examples become more powerful), t is the target label of misclassification, and the constant c can be chosen with binary search.

3.2. Generative Adversarial Networks. Generative adversarial networks (GANs) [49] are a successful framework for generative models and are widely used in many fields [50–52]. A GAN framework forces two networks to compete with each other: a generator G, which attempts to map a sample z (noise distribution $z \sim p_z(z)$ to the data distribution ($x \sim p_{\text{data}}(x)$), and a discriminative model D, which estimates the probability that a sample came from the training data rather than G. The goal of a generator G is to maximize the probability of D making a mistake. Thus, this framework plays a two-player minimax game via the following value function $V(G, D)$:

$$\begin{aligned}\min_G \max_D V(D,G) = {} & E_{x\sim p_{\text{dt}}(x)}[\log D(x)] \\ & + E_{z\sim p_z(z)}[\log(1 - D(G(z)))].\end{aligned} \tag{5}$$

In the competition, both the generator and discriminator will be improved until the discriminator cannot distinguish a generated sample from a data sample.

Mirza and Osindero [40] introduced the conditional version of generative adversarial networks (conditional GAN), and the conditional GAN can be expressed as a mapping from an observed input x and random noise z to y, G: $\{x, z\} \longrightarrow y$. The value function $V(G, D)$ in conditional GAN is as follows:

$$\begin{aligned}\min_G \max_D V(D,G) = {} & E_{x,y}[\log D(x,y)] \\ & + E_{\mathbf{x},\mathbf{z}}[log(1 - D(\mathbf{x}, \mathbf{G}(\mathbf{x},\mathbf{z})))].\end{aligned} \tag{6}$$

With the conditional GAN, it is possible to direct the data generation process and obtain the specified result.

4. Proposed Method

In this section, we introduce the defense mechanism against adversarial examples in detail.

4.1. Motivation. In computer vision, we can consider the attack and defense of adversarial examples as an image-to-image translation process. For the adversary, the goal is to perturb clean images to generate adversarial images. For the defender, the usual idea is to transform the input adversarial images and eliminate the perturbation to restore them to clean images. According to this idea, we can apply some image conversion methods to the field of adversarial examples. In 2018, Isola et al. [39] proposed a generic approach named *pix2pix* to solve image-to-image translation problems and is based on the conditional GAN. They demonstrated that *pix2pix* is effective at reconstructing objects from edge maps and colorizing images, among other tasks. In this paper, we use the same network framework as *pix2pix* to solve the problems in adversarial examples. We use the framework as a defense mechanism to generate a mapping of adversarial images to clean images.

4.2. Framework. The framework of *pix2pix* is based on the conditional GAN. This means that the structure of this framework mainly consists of two parts: a generator and a discriminator. As shown in Figure 2, we introduce the structure of our framework from two aspects.

4.2.1. Generator. We use the structure of U-Net [53] as a generator, which adds skip connections based on the encoder-decoder network. Although there are some minor distinctions in surface appearance between the inputs (adversarial images) and outputs (clean images), the underlying structures of both are the same. Therefore, in the task of image-to-image (adversarial images to clean images), both of them should share the same underlying information. The traditional encoder-decoder generator model lacks the transmission of low-level information, which causes some distortion of the outputs. Therefore, we add skip connections to share underlying information between the inputs and outputs based on the encoder-decoder network, which can ensure that the quality of the converted images is closer to the expected result. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$, where n is the total number of layers.

4.2.2. Discriminator. We use the structure of PatchGAN [39] as a discriminator. The traditional GAN discriminator judges the output as a whole, and it restricts the discriminator to model the high-frequency structure. The PatchGAN maps each input image into $N \times N$ patches via a convolutional network and attempts to determine whether each $N \times N$ patch in an image is real or fake. Then, it averages all responses to provide the ultimate output of the discriminator. In this way, the local features of the generated images can be well constructed.

4.3. Defense Adversarial Example. Figure 2 illustrates the overall architecture of the defense mechanism for the adversarial example. We use paired data (x, y) for training, and each pair of data contains a clean image y and its adversarial image x. Here, the generator G takes the adversarial example x as its input and generates the images $G(x)$. Then, $(x, G(x))$ and (x, y) are sent to the discriminator D, which is used to distinguish the generated data and

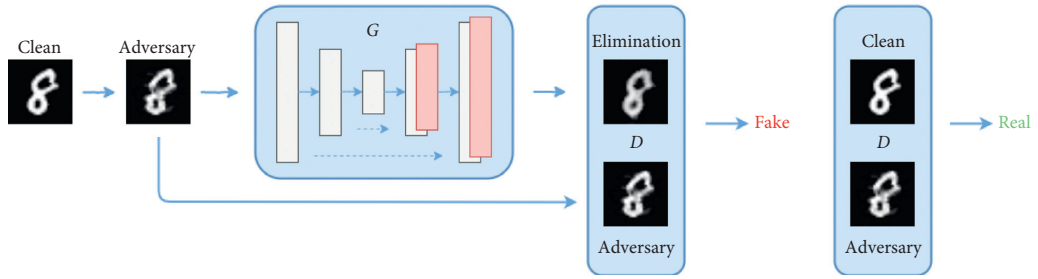

FIGURE 2: The training framework of defense mechanism with the conditional GAN. This framework consists of a generator and a discriminator. The generator takes adversarial images x as input and eliminates perturbations in x. Then, we obtain $G(x)$. The discriminator is used to distinguish the generated data $(x, G(x))$ and the original instance (x, y), where y denotes the clean images.

the original instance. The adversarial loss can be written as follows:

$$\mathscr{L}_{cGAN} = E_{x,y}[\log D(x, y)] + E_x[\log(1 - D(\mathbf{x}, \mathbf{G}(\mathbf{x})))]. \tag{7}$$

The goal of G is to not only fool the discriminator but also be near the ground truth output. Therefore, we add the loss $\mathscr{L}L_1(G)$, which encourages the generated instances $G(x)$ to be close to the clean images y:

$$\mathscr{L}L_1(G) = E_{x,y}[|y - G(x)|_1]. \tag{8}$$

The current objective function is

$$G^* = \arg\min_G \max_D \mathscr{L}_{cGAN}(G, D) + \lambda \mathscr{L}L_1(G), \tag{9}$$

where λ controls the relative importance of $\mathscr{L}L_1(G)$.

As shown in Figures 3 and 4, our defense mechanism can eliminate adversarial perturbations in the images. However, for some complex datasets (such as CIFAR10), although the generated images are close to the original clean images, their performance in the target model f is not satisfactory. To solve this problem, we adjust the objective function. Our core goal is to eliminate the adversarial perturbations in x and make the prediction results of the generated images $G(x)$ close to the prediction results of y in the target model. Therefore, we add the loss function as follows:

$$\mathscr{L}^f_{\text{adv}} = E_x L_f G(x, y). \tag{10}$$

The final objective function is

$$G^* = \arg\min_G \max_D \mathscr{L}_{cGAN}(G, D) + \lambda \mathscr{L}L_1(G) + \mu \mathscr{L}^f_{\text{adv}}, \tag{11}$$

where μ controls the relative importance of $\mathscr{L}^f_{\text{adv}}$.

In general, the loss functions $\mathscr{L}_{cGAN}(G, D)$ and $\lambda \mathscr{L}L_1(G)$ encourage the adversarial data to appear similar to the clean data, while the loss function $\mathscr{L}^f_{\text{adv}}$ improves the prediction accuracy of the generated images on the target model.

5. Experiment

In this section, we evaluate the defense mechanism against adversarial examples. All experiments are based on two datasets: MNIST and CIFAR10.

MNIST (the MNIST used to support the findings of the study is public, and one can find it in http://yann.lecun.com/exdb/mnist/) is a dataset of handwritten digits and consists of 60000 training examples and 10000 testing examples. Each sample consists of 28 × 28 pixels, where each pixel is a grayscale value. For MNIST, we trained two classifiers **Anet** and **Bnet** and used these classifiers as target models to generate adversarial examples and test our approach. The network structure is shown in Table 1. The prediction accuracies of **Anet** and **Bnet** on the test set are 98.96% and 99.74%, respectively.

The CIFAR10 (the CIFAR10 used to support the findings of the study is public, and one can find it in https://www.cs.toronto.edu/~kriz/cifar.html) dataset consists of 60000 32 × 32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. For CIFAR10, we trained two classifiers Resnet (**Rnet**) [54] and DenseNet (**Dnet**) [55] and used these classifiers as target models to generate adversarial examples and test our approach. The prediction accuracy of **Rnet** and **Dnet** on the test set is 93.63% and 95.04%, respectively.

5.1. Implementation Details. We used the adversarial examples generated by the training data and the clean images in the training data as the training set for our framework. All attacks (FGSM, DeepFool, JSMA, and CW) were implemented in *advbox* [56], which is a toolbox used to benchmark deep learning systems' vulnerabilities to adversarial examples. We used the interface provided by *advbox* to generate the adversarial examples. We experimented with $\varepsilon = 0.15$ on MNIST, $\varepsilon = 0.1$ on CIFAR10, and L_2 attacks for CW. For the targeted attacks

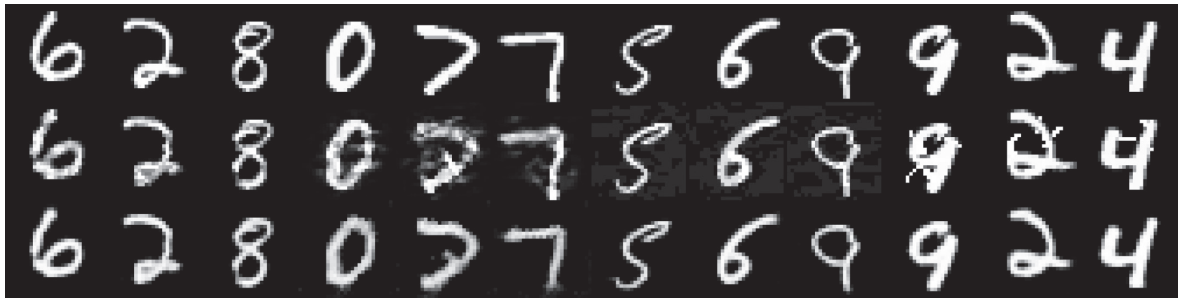

Figure 3: Experimental results of the defense adversarial example in the MNIST dataset. The first line shows clean images. The second line shows an adversarial example. Here, we use four different approaches to generate adversarial (CW, DeepFool, FGSM, and JSMA) images and three images for each approach. The third line shows the images generated by the defense framework.

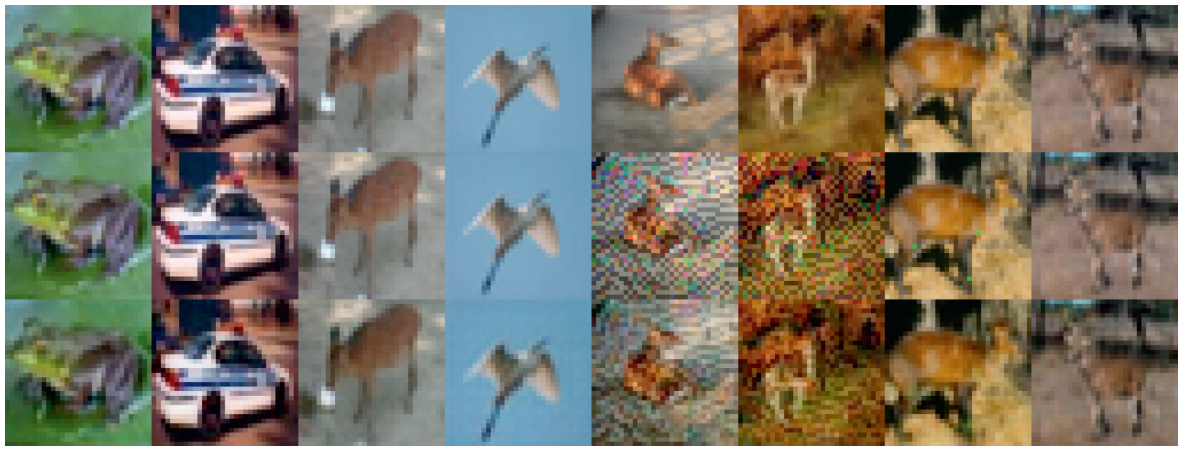

Figure 4: Experimental results of defense adversarial example in the CIFAR10 dataset. The first line shows clean images. The second line shows an adversarial example. Here, we use four different approaches to generate adversarial (CW, DeepFool, FGSM, and JSMA) images and two images for each approach. The third line shows the images generated by the defense framework.

Table 1: The structure of target model.

Anet	Bnet
Conv(64, 5×5, 1) + ReLU	Conv(64, 3×3, 1) + ReLU
Conv(64, 3×3, 2) + ReLU	Conv(64, 3×3, 1) + ReLU
Dropout(0.5)	Maxpooling
FC(128) + ReLU	Conv(64, 3×3, 1) + ReLU
Dropout(0.5)	Conv(64, 3×3, 1) + ReLU
FC(10) + ReLU	Maxpooling
SoftMax	FC(200) + ReLU
	Dropout(0.5)
	FC(200) + ReLU
	Dropout(0.5)
	FC(10) + ReLU
	SoftMax

JSMA and CW, we set a random target label for each sample. The network structure of our framework (include the generator and discriminator) is the same as *pix2pix* [39].

5.2. Defense Adversarial Example. To verify the effectiveness of the defense mechanism, we tested it on two datasets MNIST and CIFAR10. For each dataset, we trained two defense frameworks for different target models. We generated adversarial examples on test data and selected the adversarial examples that successfully attacked in the target model as members of the test set. Therefore, the prediction accuracy of the target model on the test set is 0%. In our defense mechanism, we sent the adversarial examples to a generator that had previously been trained. Then, we took the generated data as input to the target model. Figures 5 and 6 show the prediction accuracy of the target model on the adversarial example under the defense mechanism, where epoch means the number of training iterations. The result indicates that our defense framework can quickly converge during training. For the MNIST dataset, we take epoch = 20 as the final result, as shown in Table 2. Our defense mechanism is effective against different types of attacks. It improves the prediction accuracy of the target models (**Anet** and **Bnet**) on the adversarial sample from 0 to almost 98%. For the CIFAR10 dataset, we take epoch = 40 as the final result, as shown in Table 3. Since the CIFAR10 dataset is much more complicated than the MNIST dataset, it can cause some losses in the denoising process. Therefore, the defensive performance on CIFAR10 is reduced compared to that on MNIST. CW attacks are more robust than other attacks, which means that defending against such attack is more challenging. Our defense mechanism still achieves good performance on CW attacks.

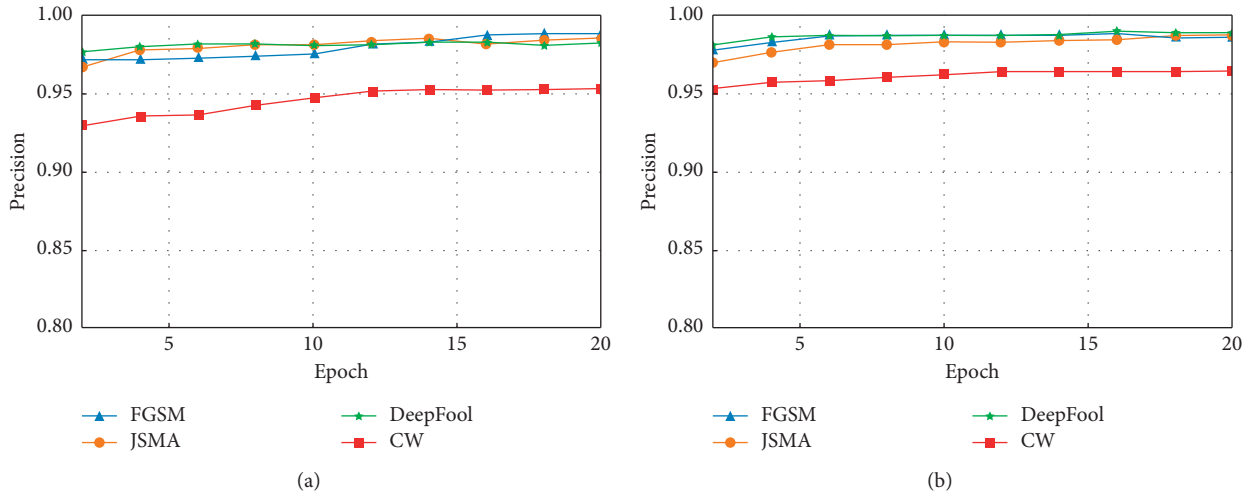

Figure 5: The prediction accuracy of the defense mechanism for the MNIST dataset (the range of epoch is from 2 to 20): the experimental results of the target models (a) Anet and (b) Bnet.

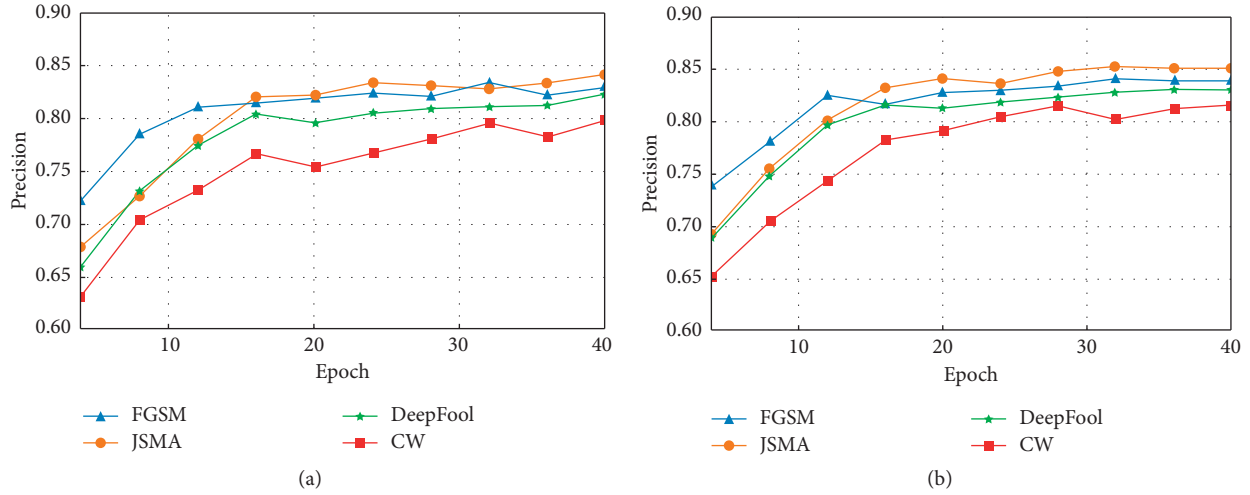

Figure 6: The prediction accuracy of the defense mechanism for CIFAR10 dataset (the range of epoch is from 4 to 40): the experimental results of target models (a) Rnet and (b) Dnet.

Table 2: Defense performance on MNIST (epoch = 20).

Target model	FGSM	DeepFool	JSMA	CW
Anet	98.73	98.44	98.25	95.39
Bnet	98.70	98.68	98.88	96.51

Table 3: Defense performance on CIFAR (epoch = 40).

Target model	FGSM	DeepFool	JSMA	CW
Rnet	82.82	84.15	82.17	79.84
Dnet	84.19	85.10	83.30	81.63

In addition, we compare the adversarial perturbation and defense loss for both the MNIST dataset (epoch = 20) and CIFAR10 dataset (epoch = 40). An adversarial perturbation means average L_1 norm loss between adversarial images and clean images, and the defense loss means an average L_1 norm loss between the generated images and

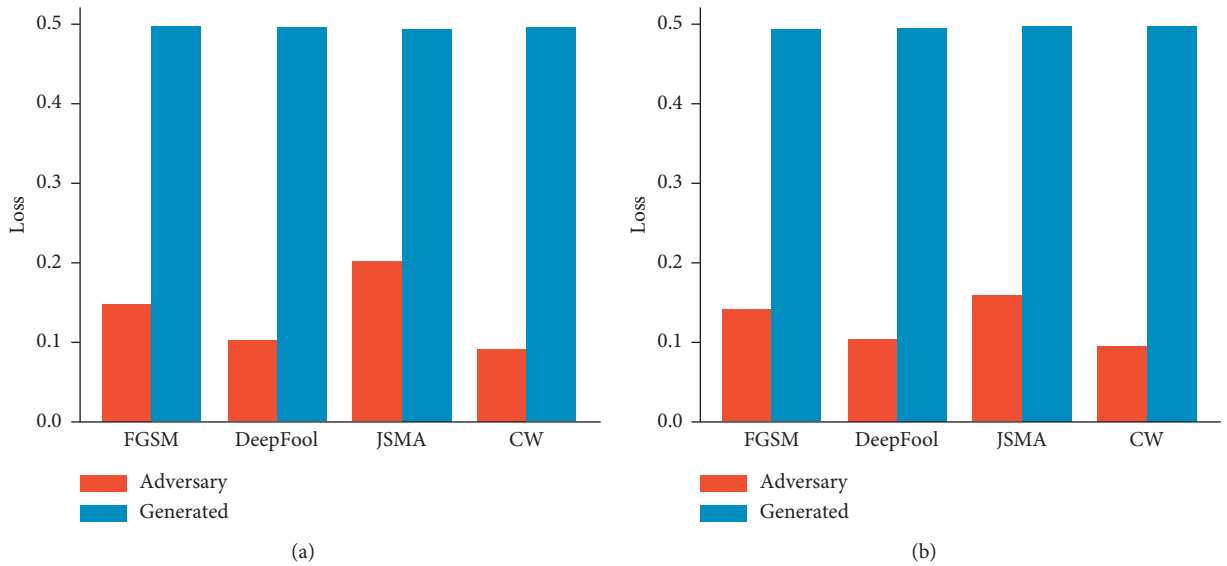

Figure 7: The loss of the defense mechanism on the MNIST dataset: experimental results of target models (a) Anet and (b) Bnet.

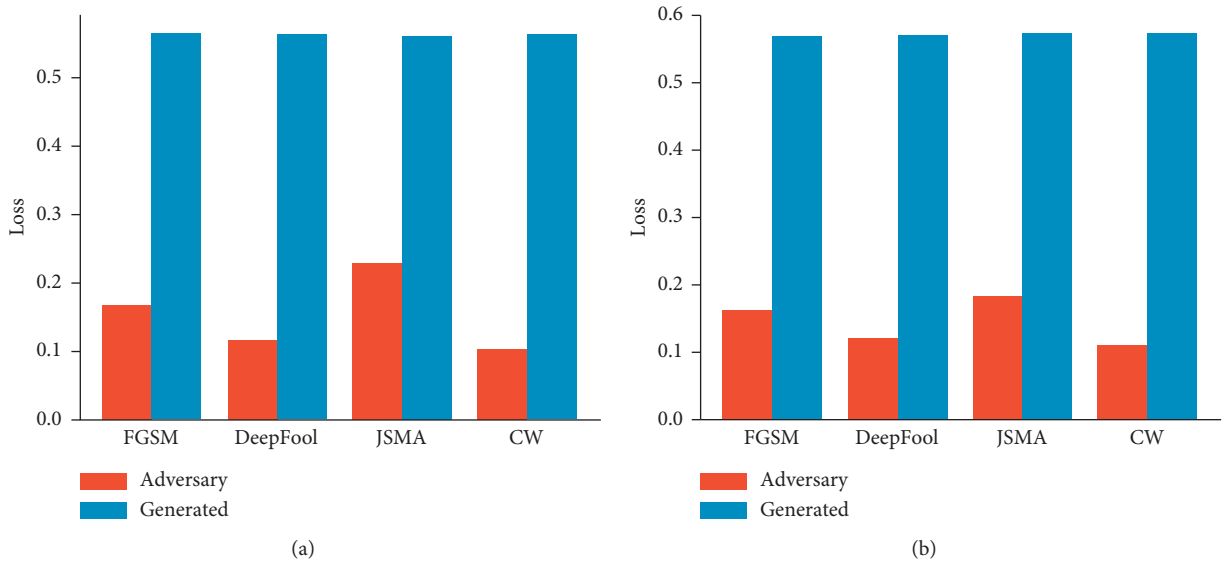

Figure 8: The loss of the defense mechanism on the CIFAR10 dataset: experimental results of target models (a) Rnet and (b) Dnet.

clean images. Since our defense framework consists of U-Net and PatchGAN, their combination enables the generator to restore the details of the original clean data. As shown in Figures 7 and 8, our defense mechanism can control defense losses within a certain range. This ensures the high quality of the generated images and the similarity to the clean images.

5.3. Defense Transferability. In this experiment, we tested the transferability of our defense mechanism. We used the adversarial examples generated by other target models to test the framework trained for the specific target model. Figures 9 and 10 show the trend of the transferability of the prediction accuracy during training. Similar to previous results, in this case, our defense mechanism still achieves a high convergence speed.

For the MNIST dataset and CIFAR10 dataset, we separately took epoch = 20 and epoch = 40 as the final result, and the result is shown in Tables 4 and 5 (**Anet**/**Bnet** means that we use the adversarial examples generated by the target model **Anet** to test the framework trained for the target model **Bnet**). The purpose of our defense mechanism is to restore the original characteristics of the adversarial examples and eliminate their adversarial perturbations. Therefore, our defense framework focuses on adversarial examples, not the target model. The experimental results prove that our method is universal. It can transfer the capabilities learned from the specific target model to other models.

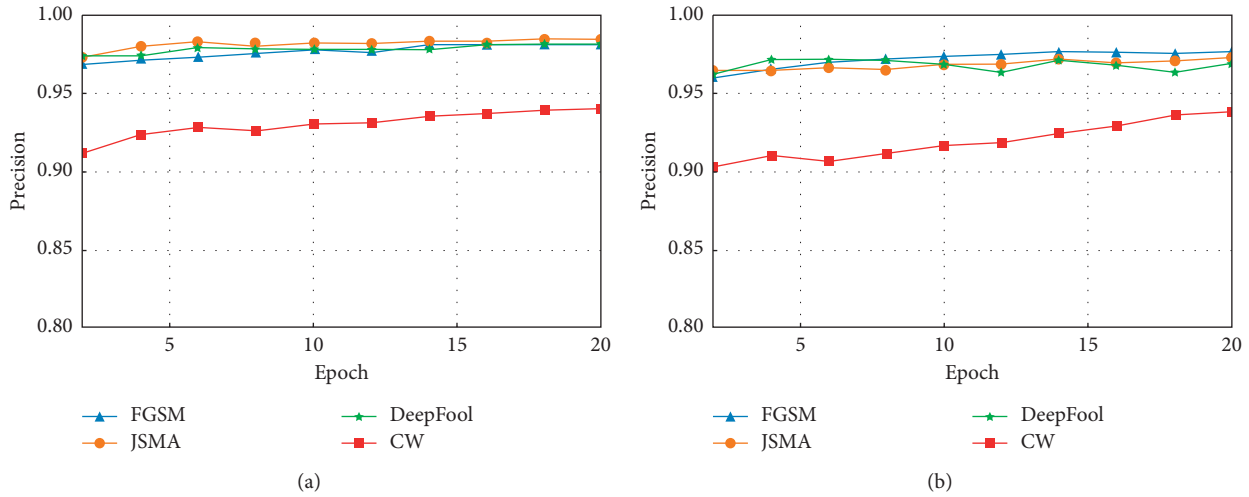

Figure 9: The transferability of the defense mechanism on the MNIST dataset (the range of epoch is from 2 to 20): experimental results of (a) Anet/Bnet and (b) Bnet/Anet.

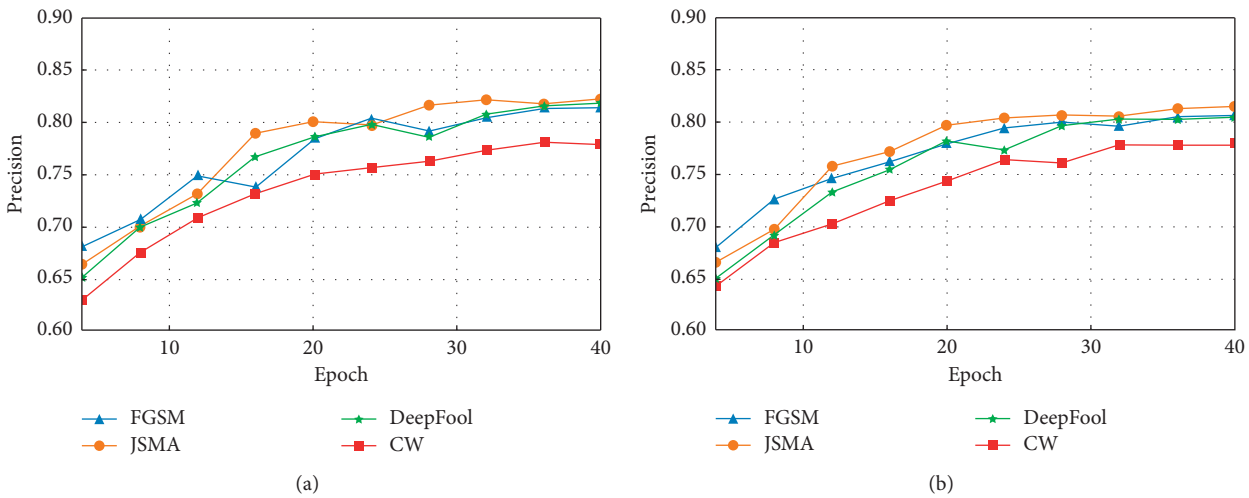

Figure 10: The transferability of the defense mechanism on the CIFAR10 dataset (the range of epoch is from 4 to 40): experimental results of (a) Rnet/Dnet and (b) Dnet/Rnet.

Table 4: Transferability on MNIST (epoch = 20).

Target model	FGSM	DeepFool	JSMA	CW
Anet/Bnet	98.13	98.36	98.06	94.11
Bnet/Anet	97.56	97.24	96.89	93.82

Table 5: Transferability on CIFAR (epoch = 40).

Target model	FGSM (%)	DeepFool (%)	JSMA (%)	CW (%)
Rnet/Dnet	81.45	82.71	81.79	78.16
Dnet/Rnet	80.36	81.36	80.35	78.01

5.4. Comparison with Other Defense Methods. Following the experimental setup in Defense-GAN [36], we compared the proposed method with other defense mechanisms such as Defense-GAN, MagNet [37], and adversarial training [27]. The adversarial training uses the adversarial example as part of the training set to build a more robust model. The magnet consists of a detector and a reformer. The detector is used to detect adversarial examples, and reformer is used to transform adversarial examples into clean examples. Since Defense-GAN is not argued secure on

Table 6: Comparisons with other defense methods.

Attack	Target model	No attack	No defense	Defense-GAN	MagNet	Adv.Tr	Our method
FGSM	A	0.997	0.217	0.988	0.191	0.651	0.986
	B	0.962	0.022	0.956	0.082	0.060	0.958
	C	0.996	0.331	0.989	0.163	0.786	0.987
	D	0.992	0.038	0.980	0.094	0.732	0.983
CW	A	0.997	0.141	0.989	0.038	0.077	0.965
	B	0.962	0.032	0.916	0.034	0.280	0.924
	C	0.996	0.126	0.989	0.025	0.031	0.968
	D	0.992	0.032	0.983	0.021	0.010	0.966

CIFAR10, we only use MNIST and experiment with $\varepsilon = 0.3$ for FGSM and the L_2 attack for CW. There are four target models A, B, C, and D, whose structures are the same as the settings in Defense-GAN. The experiment results are shown in Table 6.

The proposed method is better than MagNet and adversarial training. Although our method is slightly inferior to Defense-GAN in some tests, our method also has certain advantages. (1) Our method is simpler than Defense-GAN. Simultaneously, Defense-GAN requires two steps before feeding the input to the classifier: minimizing the reconstruction error and generating. However, our method only requires generating. (2) Our defense mechanism is a general-purpose defense framework, which means that we can adapt the defense mechanism to different datasets or scenarios with a few adjustments.

6. Conclusions

In this paper, we propose a novel defense strategy utilizing conditional GANs to enhance the robustness of classification models against adversarial examples. Our method is a universal defense framework. We tested it on different datasets and target models, and the experimental results proved that our method is effective against most commonly considered attack strategies. In addition, compared to the state-of-the-art defense methods, the proposed method also has many advantages.

It is worth mentioning that although our method is a feasible and simple defense mechanism, there are still some practical difficulties in implementing and deploying this method. For example, our experimental performance will be reduced on complex datasets. In the future, we will focus on adjusting the network structure of the defense framework to improve the performance on complex scenarios.

Data Availability

The MNIST dataset used to support the findings of the study is public and available at http://yann.lecun.com/exdb/mnist/. The CIFAR10 dataset used to support the findings of the study is public and available at https://www.cs.toronto.edu/~kriz/cifar.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61572034) and Major Science and Technology Projects in Anhui Province (18030901025).

References

[1] I. G. Goodfellow, Y. Bengio, and A. C. Courville, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[2] B. Wu, Z. Chen, J. Wang, and H. Wu, "Exponential discriminative metric embedding in deep learning," *Neurocomputing*, vol. 290, pp. 108–120, 2018.

[3] M. M. Y. Zhang, K. Shang, and H. Wu, "Learning deep discriminative face features by customized weighted constraint," *Neurocomputing*, vol. 332, pp. 71–79, 2019.

[4] C. Liu and H. Wu, "Channel pruning based on mean gradient for accelerating convolutional neural networks," *Signal Processing*, vol. 156, pp. 84–91, 2019.

[5] X. Li and H. Wu, "Spatio-temporal representation with deep neural recurrent network in mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 9, no. 5, pp. 653–657, 2020.

[6] X. Xu, R. Mo, F. Dai, W. Lin, S. Wan, and W. Dou, "Dynamic resource provisioning with fault tolerance for data-intensive meteorological workflows in cloud," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6172–6181, 2019.

[7] X. Xu, C. He, Z. Xu, L. Qi, S. Wan, and M. Z. A. Bhuiyan, "Joint optimization of offloading utility and privacy for edge computing enabled IoT," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2629–2622, 2019.

[8] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, and W. Dou, "Become: blockchain-enabled computation offloading for IoT in mobile edge computing," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4187–4195, 2019.

[9] X. Xu, X. Liu, Z. Xu, F. Dai, X. Zhang, and L. Qi, "Trust-oriented iot service placement for smart cities in edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4084–4091, 2019.

[10] X. Xu, X. Zhang, X. Liu, J. Jiang, L. Qi, and M. Z. A. Bhuiyan, "Adaptive computation offloading with edge for 5g-envisioned internet of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2020.

[11] Y. Zhang, C. Yin, Q. Wu, Q. He, and H. Zhu, "Location-aware deep collaborative filtering for service recommendation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2019.

[12] Y. Zhang, G. Cui, S. Deng, F. Chen, Y. Wang, and Q. He, "Efficient query of quality correlation for service composition," *IEEE Transactions on Services Computing*, 2018.

[13] Y. Zhang, K. Wang, Q. He et al., "Covering-based web service quality prediction via neighborhood-aware matrix factorization," *IEEE Transactions on Services Computing*, 2019.

[14] Q.-s. Zhang and S.-C. Zhu, "Visual interpretability for deep learning: a survey," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 27–39, 2018.

[15] R. Shokri, M. Stronati, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, pp. 3–18, May 2017, San Jose, CA, USA.

[16] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: analyzing the connection to overfitting," in *Proceedings of the 2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, IEEE, San Jose, CA, USA, pp. 268–282, May 2017.

[17] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic counter measures," in *Proceedings of the Computer and Communication Security*, Denver, CO, USA, October 2015.

[18] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," 2013, https://arxiv.org/abs/1312.6199.

[19] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, pp. 2805–2824, 2017.

[20] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2016, https://arxiv.org/abs/1607.02533.

[21] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Adversarial generative nets: neural network attacks on state-of-the-art face recognition," 2018, https://arxiv.org/abs/1801.00349.

[22] I. Evtimov, K. Eykholt, E. Fernandes et al., "Robust physical-world attacks on deep learning models," 2017, https://arxiv.org/abs/1707.08945.

[23] F. Trame`r, A. Kurakin, N. Papernot, D. Boneh, and P. D. McDaniel, "Ensemble adversarial training: attacks and defenses," 2017, https://arxiv.org/abs/1705.07204.

[24] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proceedings of the 2016 IEEE European Symposium on Security and Privacy*, IEEE, Saarbrucken, Germany, pp. 372–387, March 2015.

[25] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "EAD: elastic-net attacks to deep neural networks via adversarial examples," 2017, https://arxiv.org/abs/1709.04114.

[26] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, IEEE, San Jose, CA, USA, pp. 39–57, May 2017.

[27] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, https://arxiv.org/abs/1412.6572.

[28] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, https://arxiv.org/abs/1706.06083.

[29] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, NV, USA, pp. 2574–2582, June 2015.

[30] F. Trame`r, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," 2017, https://arxiv.org/abs/1704.03453.

[31] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Sri- vastava, and K.-W. Chang, "Generating natural language adversarial examples," 2018, https://arxiv.org/abs/1804.07998.

[32] Y. Qin, N. Carlini, I. Goodfellow, G. Cottrell, and C. Raffel, "Imperceptible, robust, and targeted adversarial examples for automatic speech recognition," 2019, https://arxiv.org/abs/1903.10346.

[33] F. Trame`r, A. Kurakin, N. Papernot, I. Goodfel-low, D. Boneh, and P. McDaniel, "Ensemble adversarial training: attacks and defenses," 2017, https://arxiv.org/abs/1705.07204.

[34] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy*, IEEE, San Jose, CA, USA, pp. 582–597, May 2016.

[35] X. Ma, B. Li, Y. Wang et al., "Characterizing adversarial subspaces using local intrinsic dimensionality," 2018, https://arxiv.org/abs/1801.02613.

[36] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-GAN: protecting classifiers against adversarial attacks using generative models," 2018, https://arxiv.org/abs/1805.06605.

[37] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ACM, Dallas, TX, USA, pp. 135–147, October 2017.

[38] X. Jia, X. Wei, X. Cao, and H. Foroosh, "Comdefend: An efficient image compression model to defend adversarial examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Long Beach, CA, USA, pp. 6084–6092, June 2019.

[39] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, HI, USA, pp. 5967–5976, July 2016.

[40] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[41] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. X. Song, "Generating adversarial examples with adversarial networks," in *Proceedings of the International Conference on Artificial Intelligence*, Stockholm, Sweden, July 2018.

[42] Y. Song, R. Shu, N. Kushman, and S. Ermon, "Constructing unrestricted adversarial examples with generative models," in *Proceedings of the Neural Information Processing Systems*, Montreal, Canada, December 2018.

[43] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, pp. 2642–2651, Sydney, Australia, August 2017.

[44] Z. Zhao, D. Dua, and S. Singh, "Generating natural adversarial examples," 2017, https://arxiv.org/abs/1710.11342.

[45] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, https://arxiv.org/abs/1701.07875.

[46] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on GAN," 2017, https://arxiv.org/abs/1702.05983.

[47] H. Lee, S. Han, and J. Lee, "Generative adversarial trainer: defense to adversarial perturbations with GAN," 2017, https://arxiv.org/abs/1705.03387.

[48] G. K. Santhanam and P. Grnarova, "Defending against adversarial attacks by leveraging an entire GAN," 2018, https://arxiv.org/abs/1805.10652.

[49] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2672–2680, Montreal, Canada, December 2014.

[50] H. Huang, P. S. Yu, and C. Wang, "An introduction to image synthesis with generative adversarial nets," 2018, https://arxiv.org/abs/1803.04469.

[51] C. Wang, Z. Chen, K. Shang, and H. Wu, "Label-removed generative adversarial networks incorporating with K-Means," *Neurocomputing*, vol. 361, pp. 126–136, 2019.

[52] Z. Chen, C. Wang, H. Wu, K. Shang, and J. Wang, "DMGAN: discriminative metric-based generative adversarial networks," *Knowledge-Based Systems*, vol. 192, p. 105370, 2020.

[53] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, Munich, Germany, October 2015.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, NV, USA, pp. 770–778, June 2016.

[55] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Wein-berger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Honolulu, HI, USA, pp. 4700–4708, July 2017.

[56] B. X-lab, "Advbox:a toolbox to generate adversarial examples that fool neural networks," 2019, https://github.com/baidu/AdvBox.