DEFENSE AGAINST ADVERSARIAL EXAMPLES BY ENCODER-ASSISTED SEARCH IN THE LATENT CODING SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks were shown to be vulnerable to crafted adversarial perturbations, and thus bring serious safety problems. To solve this problem, we proposed AE-GAN$_{+sr}$, a framework for purifying input images by searching a closest natural reconstruction with little computation. We first build a reconstruction network AE-GAN, which adapted auto-encoder by introducing adversarial loss to the objective function. In this way, we can enhance the generative ability of decoder and preserve the abstraction ability of encoder to form a self-organized latent space. In the inference time, when given an input, we will start a search process in the latent space which aims to find the closest reconstruction to the given image on the distribution of normal data. The encoder can provide a good start point for the searching process, which saves much computation cost. Experiments show that our method is robust against various attacks and can reach comparable even better performance to similar methods with much fewer computations.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved state-of-the-art performances among various challenging computer vision tasks. However, they are shown to be vulnerable to adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014b), which add carefully crafted perturbations to a legitimate input sample. The perturbations are small and not perceptible to a human, but they can significantly mislead the target model. Moreover, adversarial examples have considerable transferability between different models, which entertains the feasibility of black-box attacks in the real world. Therefore, it requires attention to finding effective strategies against adversarial attacks, especially black-box attacks.

There have been many efforts to defend against adversarial attacks by diminishing perturbations of samples before feeding them into classifiers. Auto-encoder based methods have achieved prevalence though (Gu & Rigazio, 2014; Meng & Chen, 2017), they are shown to be vulnerable to white-box attacks (Samangouei et al., 2018) because the model stacked by auto-encoder and classifier can be attacked again. Other generative models (Samangouei et al., 2018; Song et al., 2017) have been trained to model the distribution of normal data and project back the adversarial samples to the manifold of normal samples. In the process of purifying input images, they often include complicated optimization. Therefore, they can easily resist the white-box attack but at the cost of time.

Compared to natural images, adversarial examples have much lower probability densities under the image distribution (Song et al., 2017), which is regarded as an unknown input pattern for an auto-encoder trained with legitimate examples. As a result, the output of the encoder can be influenced by the crafted perturbations and lead to a top-down reconstruction far away from the input. When humans handle unknown input patterns, we will first find a conception in our mind and correct our conception step by step by checking the consistency between the input and the reconstruction. We can apply this mechanism to neural networks, where the latent code for reconstruction is calculated by a search process guided by feedback from the visual space. The same idea has been included in Defense-GAN (Samangouei et al., 2018), where a generator is trained to model the distribution of unperturbed images. When given an image, it uses GD minimization to find the optimum latent code, which corresponds to the closest output without perturbations to the given image. However,

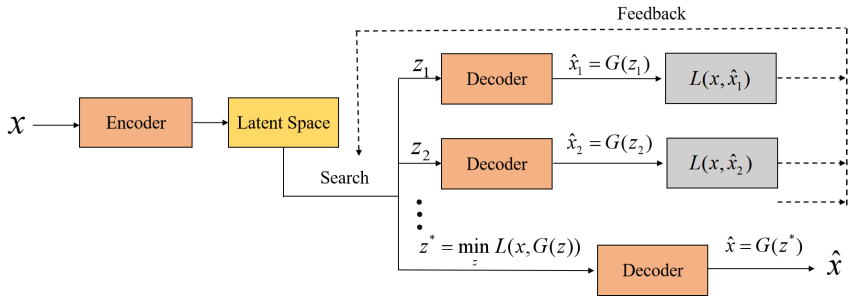


Figure 1: Framework of the encoder assisted search process. The search process is implemented in the latent space formed by the encoder and starts at the point given by the encoder. The discrepancy between the input and the reconstruction can act as the feedback from the visual space to guide the search.

Defense-GAN requires multiple random restart points and considerable iterations to guarantee the performance, which brings a high computation cost. That's because the GD minimization is sensitive to the initial points, which is given randomly in Defense-GAN.

Our approach takes the advantages of both auto-encoder and Defense-GAN. We follow the encoder-decoder framework to build the reconstruction model but also include a search process to find the best reconstruction. The encoder-decoder framework can help to form a self-organized latent space and preserves the neighborhood relations (Xu, 2019), which can provide a fair start point and a more conducive environment for the search. However, the traditional auto-encoder is trained to form an identical mapping from the input to output instead of minimizing the distance between the distribution of natural and generated images. Consequently, auto-encoder may find the closest image to a given perturbed image but out of the distribution of real data, which is still hard for a classifier to give a right prediction. To solve this problem, we add a discriminator to distinguish between natural images and generated images and force the decoder to model the distribution of natural data. The training for the reformer network is simple and requires no adversarial samples. Experimental results show that the decoder can always generate natural samples when the encoder can provide a good start point and a self-organized latent space for the search process. Besides, the encoder-decoder framework can also act as an effective detector (Meng & Chen, 2017). In the inference stage, when there comes a test sample, we first detect if the sample is from a normal distribution or not. If true, we simply reconstruct the normal image to eliminate missing noise. If not, we start the search process in the latent space, which is implemented by GD minimization, to find the closest natural reconstruction to the given image. Compared with Defense-GAN, we have two advantages: (1) The detecting enables us to discriminate normal samples and adversarial samples effectively, thus we can bypass the searching stage on normal samples, which saves much time; (2) For malicious samples, the search process starts at a good initial, thus we need no random restarts and only a few iterations. Compared with auto-encoder, we have a strong defense against white-box attacks due to the searching process. Experiments show that our method is robust against different attacks and can reach comparable performance with Defense-GAN with much fewer computations.

2 Related Work

Adversarial attacks

Various attack models have been studied to generate adversarial examples. Szegedy et al. first introduced adversarial examples against neural networks (Szegedy et al., 2013) and generated adversarial perturbations using an L-BFGS method to solve an optimization problem. For better efficiency, Goodfellow et al. proposed Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014b), an efficient single-step attack where the perturbation is updated along with the sign of the gradient of the loss w.r.t the input images. Later, Basic Iterative Method (BIM) (Kurakin et al., 2016b) is proposed as an extension of FGSM by running a finer optimization for multiple iterations and clipped pixel values in each iteration. After that, Moosavi-Dexfooli et al. proposed DeepFool (Moosavi-Dezfooli et al., 2016) to find the closest distance from the normal input to the decision boundary of adversar-

ial samples. More recently, Carli and Wagner (Carlini & Wagner, 2016) launched a powerful attack to defeat defensive distillation (Papernot et al., 2015), which generates adversarial examples with small perturbations. It is also possible to use generative models to generate universal adversarial perturbations (Reddy Mopuri et al., 2018).

DEFENSE METHODS

In recent years, a variety of techniques have been proposed to defend against adversarial attacks. Adversarial training (Kurakin et al., 2016a; Goodfellow et al., 2014b; He et al., 2017) is one of the most popular ways which trains the classifier on the augmented training set with adversarial samples. It improves the accuracy on adversarial samples but can not generalize well to new attacks (He et al., 2017). Meanwhile, it is time-consuming to generate adversarial examples and retrain the classifier. Gradient masking method (Papernot et al., 2015) has also been considered to mask or reduce the gradients in magnitude when training the classifier, but it has been shown to give a false sense of security in defending against attacks (Athalye et al., 2018). The two popular methods include retraining the classifier, while the classifier may not be allowed to be modified in the physical world.

Another way is to pre-process adversarial samples before feeding them into classifiers. Some previous work resort to traditional denoising methods (Das et al., 2017; Dziugaite et al., 2016; Osadchy et al., 2017) to purify the adversarial examples. The main drawback of these methods is that they can only fix small perturbations and may cause information loss. Denoising auto-encoder (Gu & Rigazio, 2014) is firstly proposed to defend against adversarial attacks by training an auto-encoder to map the corrupted images to clean ones. As discussed in (Samangouei et al., 2018), the stacked network is still easy to be attacked. MagNet ensembles many auto-encoders and applies a two-step defense mechanism where the hard adversarial attacks can be detected and soft adversarial attacks can be reformed. However, MagNet still suffers from white-box attacks. Different from auto-encoder, Defense-GAN (Samangouei et al., 2018) is proposed to leverage the representative power of GAN to diminish adversarial perturbations. At the inference stage, they implement a gradient descent minimization to find the best code corresponding to the closest reconstruction. Since the gradient can not pass the searching process, Defense-Gan can strongly defense against white-box attacks while the searching process is time-consuming.

3 APPROACH

3.1 THREAT MODEL

We consider the following types of adversarial attacks, according to the level of how much information is available to the attackers:

- Black-box attacks: no details about the classifier and defense mechanism are accessible. A substitute network is trained to mimic the classifier and then is used to generate adversarial examples.
- White-box attacks: complete access to all the information about the classifier and the defense strategy.
- Gray-box attacks: complete access to the classifier but no access to defense strategy.

3.2 METHOD OVERVIEW

Let $f_\theta(x) : \mathbb{R}^d \to \mathbb{N}$ be a classifier parameterized by θ, where $\mathbb{R}^d$ is the image space and $\mathbb{N}$ is the set of natural numbers denoting class labels. Given a clean image x, an adversary can perturb the clean image with small perturbation η but confuse f:

$$f_\theta(x + \eta) \neq f_\theta(x), \|\eta\| < \epsilon_{attack} \tag{1}$$

where $\|\cdot\|$ is a measurement and ϵ_{attack} is the perturbation scale which sets the maximum perturbation allowed for each pixel and often set to a small number to get almost imperceptible difference between $x_{adv} = x + \eta$ and x.

We want to improve robustness by learning a transformation $T(.)$, such that the predicted label of the transformed image does not change when the image corrupted with perturbations, i.e.,

(a) (b) (c)

Figure 2: Purified images on (a) MNIST and (b) F-MNIST dataset, under FGSM ($\epsilon = 0.3$) gray-box attack, as well as (c) Half-Masked attack on F-MNIST. First row: the corrupted images; Second row: purified image by $\text{AE}_{+\text{s}}$; Third row: purified image by $\text{A.G.}_{+\text{s}}$; Last row: ground truth.

$f_\theta(T(x_{adv})) = f_\theta(T(x))$. The basic idea of our method is to purify input images by searching the closest reconstruction in the training distribution within little computation cost. To achieve this, we build a reconstruction network based on the encoder-decoder framework, which is denoted as AE-GAN and will be described in detail in Section 3.3. After training, the encoder of AE-GAN can provide a good representation for most inputs, and the decoder can always generate samples of normal distribution during the search.

In the inference time, when given an input, we first detect if the sample is from a normal distribution or not. If true, we simply reconstruct the normal image to diminish missing noises. If not, we start a search process in the latent coding space to find the best code, which corresponds to the closest reconstruction without perturbations. The search process starts at the output of the encoder and implemented by GD minimization, which will be described in Section 3.4.

3.3 Architecture of AE-GAN

An auto-encoder (AE) is composed of two components: the encoder F and the decoder G. Let $\mathcal{X}$ be the visual space and $\mathcal{Z}$ be the latent space. The mapping $F : \mathcal{X} \to \mathcal{Z}$ performs image abstraction, which encode the image $x \in \mathcal{X}$ into a code $z \in \mathcal{Z}$. The mapping $\mathcal{Z} \to \mathcal{X}$ performs image generation, which generates image $x \in \mathcal{X}$ from the output of encoder $z \in \mathcal{Z}$. Formally, we denote the distribution of training images as p_{data}. The loss function of AE is defined as:

$$\mathbb{E}_{x \sim p_{data}} \|G(F(x)) - x\| \tag{2}$$

As shown in Equation 2, traditional AE minimizes the euclidean distance between input and output. Thus, it is trained to form an identical mapping from the input to output instead of minimizing the distance between the distribution of natural images p_{data} and generated images p_g. Consequently, when we search for the best code z in the latent space, AE may find a close image $G(z)$ but out of P_{data}, as shown in Figure 2, which is a meaningless purification because the accuracy of classifier on these images is not ideal. To solve this problem, we adapt the auto-encoder by introducing the min-max loss in GANs (Goodfellow et al., 2014a). GANs consists of two neural networks, a generator G and a discriminator D. G maps a low-dimensional latent space $\mathcal{Z}$ to the high dimensional sample space $\mathcal{X}$. D is a binary neural network classifier. While G learns to generate samples that similar to the real data x, D learns to distinguish between "real" samples x and "fake" samples $G(z)$. D and G are trained in an alternating fashion to minimize the following min-max loss (Goodfellow et al., 2014a):

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z))] \tag{3}$$

To preserve the advantages of AE, we still keep the mean square error to drive the abstraction mapping and self-organising in the latent space. Meanwhile, we add a discriminator learning to distinguish between the samples generated by the decoder and real samples. Thus, our final objective

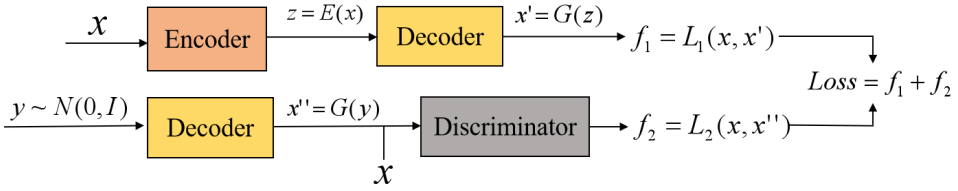


Figure 3: Architecture of AE-GAN

is to minimize the following min-max loss:

$$\min_{F,G}\max_{D} V(D,F,G) = \mathbb{E}_{x\sim p_{data}}[log(D(x))] + \mathbb{E}_{z\sim p_z}[log(1-D(G(z)))] + \mathbb{E}_{x\sim p_{data}}\|G(F(x)) - x\| \tag{4}$$

As the start point of the search process, the output of the encoder is expected to be close to the optim code and thus assist the search to converge fastly. Since AE-GAN is trained with normal images, the encoder F may suffer from the covariate shift when the input is corrupted with large perturbations. To solve this, when calculating the reconstruction error, the clean image is perturbated with random Gaussian noise before feeding into the encoder. Adding noise in the input layer acts as a regularization (Bishop, 1995), thus we denote the method as noise regularization ($+r$) in this paper and the final objective function of AE-GAN$_{+\text{r}}$ can be written as Equation 5.

$$\min_{F,G}\max_{D} V(D,F,G) = \mathbb{E}_{x\sim p_{data}}[log(D(x))] + \mathbb{E}_{z\sim p_z}[log(1-D(G(z)))] + \mathbb{E}_{x\sim p_{data}}\|G(F(x+\eta)) - x\| \tag{5}$$

where $\eta \sim N(0,\sigma)$, σ follows the uniform distribution on a given region.

Bidirectional Generative Adversarial Networks (BiGANs) (Donahue et al., 2016) also includes encoder-decoder mapping. In addition to the generator from the standard GAN framework, BiGAN includes an encoder to map the input data into the latent representation. The discriminator in BiGAN discriminates jointly data x and latent code $E(X)$ (tuples $(x, E(x))$ versus $(G(z), z)$), where the latent code is either an encoder output (for real samples) or a generator input z (for generated samples).

3.4 Encoder assited search process

Although AE-GAN$_{+\text{r}}$ has been trained with noise regularization, the added noise is simple while the adversarial noise is carefully crafted, the distribution between them is different. Moreover, when the input x^* is seriously corrupted, e.g., a large region of the image is masked, the code derived from encoder $F(x^*)$ can hardly represent the original image x. Therefore, a correction for $F(x^*)$ is required when the input pattern is significantly different from the training data. Meanwhile, the end-to-end encoder-decoder framework can not perform well against white-box attacks (Samangouei et al., 2018). Therefore, we include a search (denoted as $+s$) process in the latent space.

As shown in Figure 1, the core idea is that for every input sample x, AE-GAN$_{+\text{s}}$ will process a search process in the latent space to find the best code for an input, which can lead the decoder to generate the closest reconstruction. For a given input x, we define the objective function J of the search process as:

$$J(z) = L(x, G(z)), \text{for } z \in \mathcal{Z} \tag{6}$$

where $G(.)$ denotes the decoder, $L(.)$ is a measurement of the discrepancy between x and $G(z)$. We aims to find the optimim code z^* such that

$$z^* = \arg\min_{z\in\mathcal{Z}} J(z) \tag{7}$$

Since our latent space is designed to be continuous, thus the direction and step size can be calculated by Gradient Decent (GD) method. Therefore, the search process is converted to a GD minimization to minimize the discrepancy between the input and the reconstruction. Here we simply use euclidean distance on the pixel level to measure the dicrepancy.

In our framework, the encoder forms a self-organized latent space which preserves neighbourhood relations and provides a good start point. Both of them can assist the search process in converging. In Defense-GAN (Samangouei et al., 2018), the latent space follows a single Gaussian distribution while the gradient descent method is sensitive to initial values. Therefore, it requires multiple random restart points and considerable iterations to guarantee the performance, which brings a high computation cost.

4 EXPERIMENTS

4.1 SETTINGS

In this section, we carry out our experiments on two image datasets: the MNIST (LeCun et al., 1998) and F-MNIST (Xiao et al., 2017) dataset. Both of them contain $60,000$ training images and $10,000$ testing images. We split the original training set by 20:1 for training and validation, and keep the original testing set as our testing set. Besides, we randomly choose 2000 samples from the testing set as a development dataset, shortly dev-set, on which we conduct some exploratory experiments. For the architecture and training of the target classifier A, we follow the settings in Defense-GAN(Samangouei et al., 2018) and the details are described in Appendix B. After training, we get an accuracy of 99.6% and 91.8% on MNIST and F-MNIST respectively, which near state of the art. For the architecture and training details of AE-GAN, we adopt the settings in Defense-GAN (Samangouei et al., 2018) and restate them in Appendix B with a detailed description.

Specifically, we consider the white-box and gray-box attacks by FGSM (Goodfellow et al., 2014b), PGD (Madry et al., 2017) and Carlini-Wagner (CW) attack with ℓ_2 norm (Carlini & Wagner, 2016), and black-box attacks by FGSM. The training procedure of the substitute model in black-box attacks is the same as the setting in (Papernot et al., 2017). We split the testing set into a small hold-out set of 150 samples for training the substitute model and the remaining 9850 samples for testing different defense strategies. For all the attack methods, we use the implementation in Cleverhans (Papernot et al., 2018), a python library to benchmark machine learning systems' vulnerability to adversarial examples, and enforce the image to remain within $[0, 1]^{H\times W}$ by clipping.

4.2 RESULTS ON ADVERSARIAL ATTACKS

We compare our method to adversarial training (Szegedy et al., 2013; Goodfellow et al., 2014b), MagNet(Meng & Chen, 2017), Defense-Gan(Samangouei et al., 2018), Bi-GAN(Donahue et al., 2016) under the FGSM, PGD and CW(with l_2 norm) white-box and gray-box attacks, as well as the FGSM black-box attack. Our reported method is AE-GAN$_{+\text{sr}}$, i.e., AE-GAN with nosie regularization and search process, where the number of iterations $L = 15$. We also assist Bi-GAN with the search process for fair comparisons, which is denoted as Bi-GAN$_{+\text{s}}$. Reported Defense-GAN follows the original settings where iterarions $L = 200$ and restart points $R = 10$. All experimental results are summarized in Table 1.

From Table 1, we can observe that adversarial training successfully defends against the FGSM attack, but can not generalize to other attacks, that is because adversarial training requires adversarial samples to re-train the classifier. As discussed in Section 3.3, on small scale perturbations like CW the performance of MagNet and AE-GAN$_{+\text{sr}}$ is close, because the code derived from the encoder is robust to small perturbations. However, on large scale perturbations like FGSM ($\epsilon = 0.3$) and PGD ($\epsilon = 0.3$), the reported AE-GAN$_{+\text{sr}}$ can achieve about $20\% \sim 30\%$ improvements than MagNet.

As shown in Table 1, adversarial training and MagNet can only perform well under specific situations, while Defense-GAN and our method can defense against various attacks. Attack-agnostic is important for a defense strategy because we have no information about the attacker in the physical world. While our method reaches comparable performance with Defense-GAN under white-box and black-box attacks, we significantly reduce the computation complexity to $15/2000$ of Defense-GAN. Another observation is that when the performance both decreases on the F-MNIST dataset due to the large perturbations, our method performs much better than Defense-GAN on gray-box attacks. The possible reason is that images in F-MNIST are more complex and require more details, which raises the difficulty of searching the best code in the latent space from random points. For white-box attacks, the performance of our method has a slight drop from Defense-GAN. However, it has been studied in (Samangouei et al., 2018) that more iterations can increase the robustness

Table 1: Classification accuracies of classifier A using various defense strategies on the MNIST (top) and F-MNIST (bottom) dataset, under FGSM ($\epsilon = 0.3$), PGD ($\epsilon = 0.3$) and CW (with L_2norm) white-box attacks, gray-box attacks, and FGSM black-box attacks. Here the corresponding iterations of GD minimization are: 15 for our method, 60 for BiGAN, and 200 for Defense-GAN. (Adv.Tr.: adversarial training with FGSM ($\epsilon = 0.3$))

Attack	Method	None	Our	Adv. Tr.	MagNet	Bi-GAN$_{+s}$	Defense-Gan
White	FGSM	0.144	**0.984**	0.651	0.293	0.912	0.981
	PGD	0.007	0.982	0.059	0.012	0.924	**0.989**
	CW	0.008	0.979	0.007	0.013	0.905	**0.981**
Gray	FGSM	0.144	**0.882**	-	0.561	0.731	0.880
	PGD	0.007	0.910	-	0.673	0.739	**0.913**
	CW	0.008	0.915	-	0.881	0.805	**0.918**
Black	A/B	0.701	0.935	**0.951**	0.464	0.701	0.928
	A/E	0.813	0.933	**0.963**	0.590	0.632	0.922

Attack	Method	None	Our	Adv. Tr.	MagNet	Bi-GAN$_{+s}$	Defense-Gan
White	FGSM	0.073	0.804	0.797	0.199	0.613	**0.814**
	PGD	0.028	0.816	0.150	0.071	0.653	**0.852**
	CW	0.062	0.767	0.157	0.084	0.601	**0.810**
Gray	FGSM	0.073	**0.607**	-	0.304	0.570	0.389
	PGD	0.028	**0.736**	-	0.427	0.605	0.491
	CW	0.062	**0.719**	-	0.646	0.608	0.435
Black	A/B	0.227	0.603	**0.769**	0.308	0.398	0.602
	A/E	0.295	0.507	**0.694**	0.267	0.348	0.461

for Defense-GAN against GD-based white-box attacks. When we decline L from 200 to 15 for Defense-GAN, the performance drops by 8% decline.

4.3 Trade-off between Computation and Performance

We further investigate the trade-off between computation cost and performance in Defense-GAN and our method. Tabel 2 shows the robustness of Defense-GAN and AE-GAN$_{+sr}$ when changing the number of restart points R and iterations L. It can be seen that with less restart points, the performance of Defense-GAN decreases a lot after a certain L value, which means that the effect of varying R is extremely pronounced. This is due to the non-convex nature of MSE and increasing R enables Defense-GAN to sample different local minimum (Samangouei et al., 2018). However, in our framework, taking the output of the encoder as the initial start point can enable the search process to reach a reasonable local minimum in a few iterations. When we decrease L from 60 to 10, there is $6\% - 10\%$ drop for the accuracy of Defense-GAN, while there is little change on the performance of AE-GAN$_{+sr}$. Therefore, compared with Defense-GAN, our method requires less computation than Defense-GAN when they reach comparable performance. The performance of our is robust to the computation.

4.4 Attack Detection

AE-GAN$_{+r}$ can well reconstruct natural images without searching. From this observation, we propose to introduce the detection mechanism to bypass the searching process for clean images. In AE-GAN$_{+r}$, the decoder is trained to produce images which resemble the legitimate data. Thus, the adversarial examples usually have a high reconstruction error. Therefore, we can choose the reconstruction error as our indicator to decide whether or not the image is adversarial. The setting for the threshold is important for the detection. A larger threshold can tolerant more perturbations

Table 2: Classification accuracy of classifier A using different defense strategy on test set of MNIST (left) and F-MNIST (right), under FGSM ($\epsilon = 0.3$) attack, with different GD iterations $L = 10, 60, 200$ and various numbers of restart points $R = 1, 10$.

Attack	Method	$R = 1$		$R = 10$	
		$L = 10$	$L = 60$	$L = 10$	$L = 60$
Clean	Our	**0.982\0.837**	**0.985\0.841**	-	-
	Defense-Gan	0.545\0.405	0.693 \0.530	0.909 \ 0.696	0.967\ 0.788
White	Our	**0.984\0.799**	**0.985\0.804**	-	-
	Defense-Gan	0.537\ 0.403	0.688\0.527	0.905\0.697	0.964\0.787
Gray	Our	**0.880 \0.604**	**0.884\0.595**	-	-
	Defense-Gan	0.398\0.296	0.570 \0.325	0.769\ 0.441	0.863\0.432

and enhance efficiency but is more unsafety. Here we propose some effective methods for calculating a proper θ according to demand using the statistical information from the normal images. For example,

$$\theta_1 = \arg\min_{\theta \in \Theta_{90}} (x - G(F(x))^2) = 0.025, \quad \theta_2 = E_{x \sim p_{data}} (x - G(F(x)))^2 = 0.015$$

where $\Theta_\alpha = \{\theta : (x - G(F(x)))^2 < \theta \text{ holds for more than } \alpha\% \text{ samples}\}$. Using θ_1 as the threshold means that the detector can filter most normal samples and tolerant for small perturbations, while θ_2, i.e., the average reconstruction error on the training dataset, is more strict.

In Figure 4, we visualize the detection performance of different thresholds. The three dotted lines are the changing of detection accuracies w.r.t the strength of attacks using different thresholds. Their intersections with the performance line of AE-GAN$_{+r}$ (the solid black line) are the conservative performances of corresponding defence strategies. We can also observe that, for small perturbations missing detection, AE-GAN$_{+r}$ can successfully eliminate them. For large perturbations, they will be detected, and the search process will start. Actually, the three thresholds correspond to three strategies: high accuracy (orange), high efficiency (blue), and the medium (red). This enables the framework flexible to different demand.

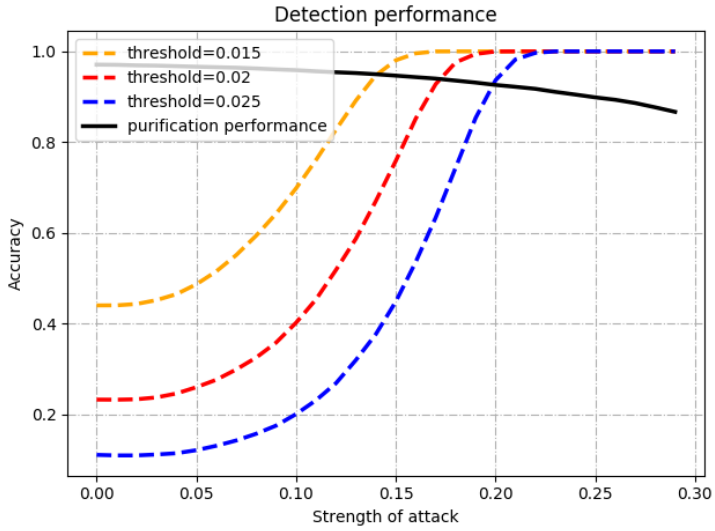


Figure 4: Defense performance on different strength of FGSM attacks on MNIST dataset. Dotted lines are detection performance, where the performance is measured as the percentage of adversarial examples are detected by the detector. The solid black line is the accuracy of classifier A on the image purifed by AE-GAN$_{+r}$.

Table 3: Classification accuracies of classifier A using various defense strategies on the MNIST dev-set, under FGSM ($\epsilon = 0.3$), CW (with l_2 norm) gray-box attack and Half-Masked (H.M.) attack. (A.G.: AE-GAN; B.G.: Bi-GAN; +s: includes search process; +r: with noise regularization.)

D	AE	AE_{+s}	B.G.	$B.G._{+s}$	A.G.	$A.G._{+s}$	$A.G._{+r}$	$A.G._{+sr}$
FGSM	0.515	0.678	0.428	0.688	0.502	0.781	0.806	**0.832**
CW	0.861	0.852	0.477	0.749	0.868	0.887	0.870	**0.889**
H.M	0.552	0.672	0.408	0.590	0.494	**0.807**	0.570	0.776

4.5 Ablation Study

To investigate the effect of each module in our method, we compare the performance of different models under the FGSM gray-box attack and half-masked attack (half of the input image is blocked, as shown in Figure 2(c)). Table 3 shows the classification performance of the different models on different image corruptions visualized in Figure 2, where the number of iterations L is set to be 20 for models with search process.

Firstly, by comparing the performance of AE-GAN and AE-GAN_{+r}, we can observe that noise regularization has an expected effect of forming an accurate latent code for the corrupted input. Secondly, the accuracies of AE_{+s} and AE-GAN_{+s} as well as the qualitative results in Figure 2 reveals that the introduced discriminator and adversarial loss improve the generative ability of the decoder. Thirdly, we also look at the consistency between input and output of Bi-GAN. As illustrated in Table 3, the accuracy of Bi-GAN is not ideal which indicates that the output of the encoder is sometimes inconsistent with the original images, thus the start point can not assist the searching process in Bi-GAN_{+s} to reach a local minimum within a certain number of steps.

5 Conclusion

We proposed AE-GAN_{+sr}, a framework to effectively defense against various adversarial attacks. Our method takes the advantages of both auto-encoder and GANs and avoids their drawbacks. We build a reconstruction network following the encoder-decoder framework, as well as introduce the search process in the latent space to find the closet reconstruction without perturbations to the given image. Compared with auto-encoder based methods, we can strongly defend against white-box attacks. Compared with GANs method, which requires multiple random restart points and considerable iterations to purify a sample, the encoder in our framework can form a self-organised latent space and provide a good start point for the search process, which enables the search converge fastly. We empirically show that we significantly reduce the computation cost while we reach comparable even better performance compared with the GANs method. The training for our network requires no adversarial examples. Meanwhile, our detection mechanism can bypass the search process for clean images, which can further improve efficiency.

References

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *CoRR*, abs/1802.00420, 2018. URL `http://arxiv.org/abs/1802.00420`.

Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *CoRR*, abs/1608.04644, 2016. URL `http://arxiv.org/abs/1608.04644`.

Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference, 2016.

Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014a.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17)*, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016a.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016b. URL http://arxiv.org/abs/1607.02533.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017.

Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. *CoRR*, abs/1705.09064, 2017. URL http://arxiv.org/abs/1705.09064.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Margarita Osadchy, Julio Hernandez-Castro, Stuart Gibson, Orr Dunkelman, and Daniel Pérez-Cabo. No bot expects the deepcaptcha! introducing immutable adversarial examples, with applications to captcha generation. *IEEE Transactions on Information Forensics and Security*, 12(11): 2640–2653, 2017.

Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. *CoRR*, abs/1511.04508, 2015. URL http://arxiv.org/abs/1511.04508.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pp. 506–519, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4944-4. doi: 10.1145/3052973.3053009. URL http://doi.acm.org/10.1145/3052973.3053009.

Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 742–751, 2018.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL http://arxiv.org/abs/1312.6199.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

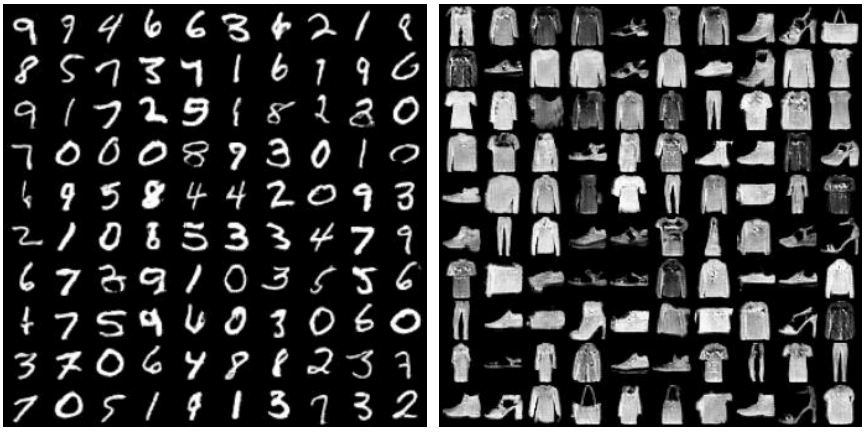
Lei Xu. An overview and perspectives on bidirectional intelligence: Lmser duality, double ia harmony, and causal computation. *IEEE/CAA Journal of Automatica Sinica*, 6(4):865–893, 2019.

A IMPLEMENTATIONS

For adversarial training, we follow the settings in Defense-GAN(Samangouei et al., 2018). We obtain adversarial examples from FGSM with $\epsilon = 0.3$, which is consistent with the strength in the inference stage.

For Defense-GAN, we adopt the implementation of original work (Samangouei et al., 2018).

For Bi-GAN, the original implementation (Donahue et al., 2016) is based on the fully-connected network. For fair comparisons, we adopt the convolutional architecture in another similar work (Dumoulin et al., 2016). We slightly adapt the architecture on CIFAR-10(Krizhevsky et al., 2009) by changing the input channel from 3 to 1. As shown in Figure 5, the model we trained reach a fair performance on generating samples.



(a) (b)

Figure 5: Generated samples by Bi-GAN trained on MNIST (left) and F-MNIST (right).

For MagNet, we follow the implementation in Defense-GAN (Samangouei et al., 2018) and restate the details in Table 6, which shares the same architecture with AE-GAN. However, we find that the latent space dimension 128 is too large for MagNet to eliminate strong perturbations. Because the details of noise are also encoded and can be reconstructed. Thus, we also take the latent dimension as 20 for comparisons. The results in Table 8 show that the change can enhance the performance on the gray-box attack.

B NEURAL NETWORK ARCHITECURES AND TRAINING SETTINGS

For fair comparisons, we compare different strategies through this paper on the same classifier A, and we take two substitute classifiers for balck-box attacks, B for convolutional structure and E for fully connected structure. The training settings and architectures almost follow the settings in Defense-GAN(Samangouei et al., 2018). For clarification, here we restate their architectures and training settings in Table 5 and Table 7 respectively.

The architecture of our model and MagNet is described in Table 6, which make a slight modification on the MagNet structure given in (Samangouei et al., 2018)

Table 4: Training parameters for AE-GAN and MagNet

Parameters	MNIST, F-MNIST
Epochs	60
Learning Rate	0.0002
Optimization Method	Adam
Batch Size	100

Table 5: Neural architectures used for classifiers and substitute models

A	B	C
Conv(64, 5×5, 1)	Conv(64, 8×8, 2)	
ReLU	ReLU	FC(200)
Conv(64, 5×5, 2)	Conv(128, 6×6, 2)	ReLU
ReLU	ReLU	Dropout(0.5)
Dropout(0.25)	Conv(128, 5×5, 1)	FC(200)
FC(128)	ReLU	ReLU
ReLU	Drpupout(0.5)	Dropout(0.5)
Drpupout(0.5)	FC(10)+Softmax	FC(10)+Softmax
FC(10)+Softmax		

Table 6: Neural architectures used for AE-GAN and MagNet

Encoder	Decoder	Discriminator
Conv(1, 64, 5×5, 2)	FC(1024) & ReLU	Conv(1, 64, 5×5, 2)
LeakyReLU(0.2)	ConvT(256, 128, 5×5, 2)	LeakyReLU(0.2)
Conv(64, 128, 5×5, 2)	BatchNorm & ReLU	Conv(64, 128, 5×5, 2)
BatchNorm & LeakyReLU	ConvT(128, 64, 5×5, 2)	BatchNorm & LeakyReLU
Conv(128, 256, 5×5, 2)	BatchNorm & ReLU	Conv(128, 256, 5×5, 2)
BatchNorm & LeakyReLU	ConvT(64, 1, 5×5, 2)	BatchNorm & LeakyReLU
FC(128)+tanh	Sigmoid	FC(1)+Sigmoid

Table 7: Training parameters for classifiers

Parameters	MNIST, F-MNIST
Epochs	10
Learning Rate	0.001
Optimization Method	Adam
Batch Size	100

C ADDITIONAL RESULTS ON VARIOUS LATETN DIMENTION

Table 8: Classification accuracy of A using MagNet and AE-GAN on MNIST with various latent space dimension. D: the dimension of latent space. the number of iterations is set to 15 for

Attack	Attack	None	D=20		D=128	
			MagNet	AE-GAN$_{+sr}$	MagNet	AE-GAN$_{+sr}$
White	FGSM	0.144	0.298	0.972	0.293	0.984
	PGD	0.007	0.012	0.975	0.006	0.982
	CW	0.008	0.013	0.970	0.008	0.979
Gray	FGSM	0.144	0.561	0.875	0.316	0.882
	PGD	0.007	0.673	0.911	0.145	0.910
	CW	0.008	0.881	0.907	0.345	0.915

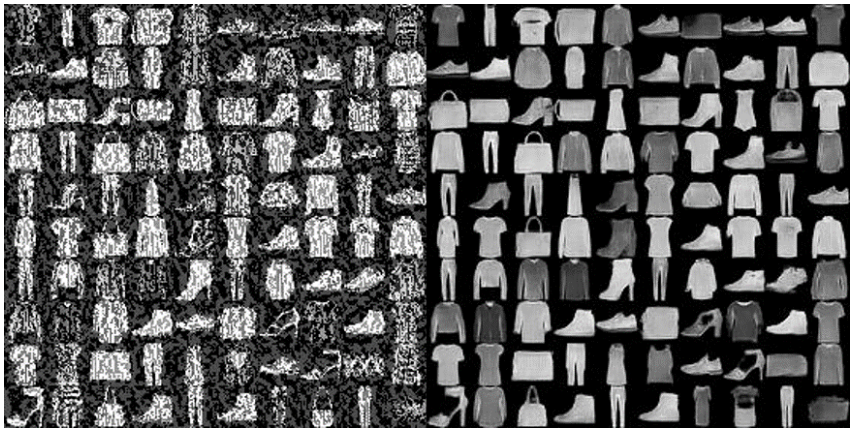
D QUALITATIVE EXAMPLES

(a)

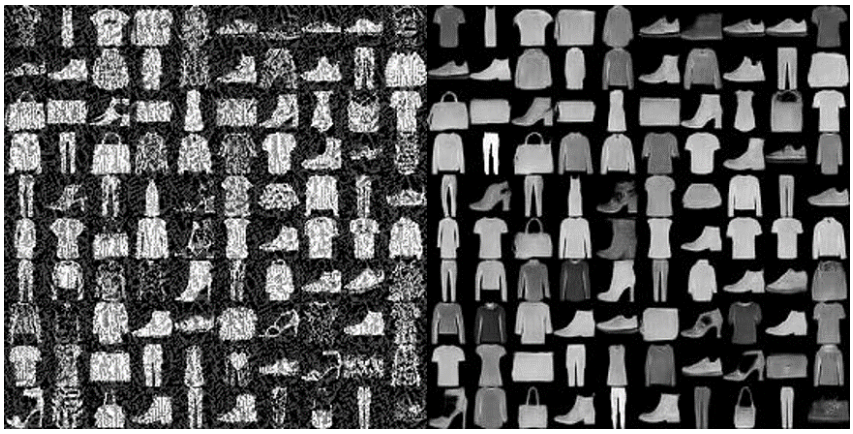
(b)

(c)

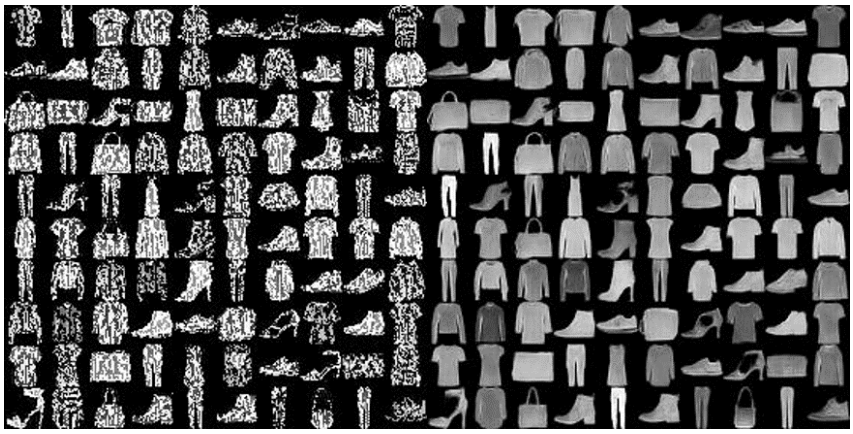
Figure 6: Example results of our method on MNIST. The left part shows adversarial examples generated by different attacks: FGSM attack with $\epsilon = 0.3$ (a), PGD attack with $\epsilon = 3$ (b) and CW with l_2 norm (c), while the right part shows purified images by AE-GAN$_{+\text{sr}}$. The number of iterations for guided search is set to 15.



(a)



(b)



(c)

Figure 7: Example results of our method on F-MNIST. The left part shows adversarial examples generated by different attacks: FGSM attack with $\epsilon = 0.3$ (a), PGD attack with $\epsilon = 3$ (b) and CW with l_2 norm (c), while the right part shows purified images by AE-GAN$_{+\text{sr}}$. The number of iterations for guided search is set to 15.